

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

THIS PAGE BLANK (USPTO)

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-036529
(43)Date of publication of application : 07.02.1995

(51)Int.Cl. G05B 19/4155
G06F 9/46

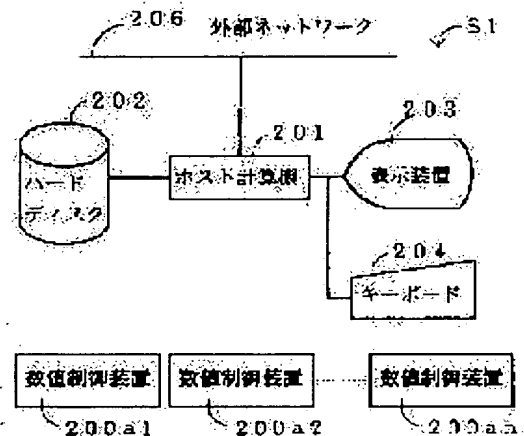
(21)Application number : 05-175490 (71)Applicant : MITSUBISHI ELECTRIC CORP
(22)Date of filing : 15.07.1993 (72)Inventor : NAMIKADO SHIGEKI

(54) EXECUTION SYSTEM FOR CONTROL SOFTWARE OF NUMERICAL CONTROLLER

(57)Abstract:

PURPOSE: To effectively carry out the control software at a low cost by allocating the task functions of a numerical controller to the threads in a process or the tasks which are carried out in an operating system of the general-purpose multitask or multithread.

CONSTITUTION: A control software execution system S1 consists of the numerical controllers 200a1-200an, a host computer 201, a hard disk 202, a display device 203, and a keyboard 204. The controller 200a1-200an carry out the control software in a real-time operating system and attain the functions of the numerical controllers. The computer 201 carries out the control software of the numerical controllers in a general-purpose multithread operating system. The disk 202 functions as an external storage of the computer 201, and the device 203 shows various necessary information for users.



LEGAL STATUS

[Date of request for examination] 05.12.1996
[Date of sending the examiner's decision of rejection]
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]
[Patent number] 2820189
[Date of registration] 28.08.1998
[Number of appeal against examiner's decision of rejection]
[Date of requesting appeal against examiner's decision of rejection]
[Date of extinction of right]

THIS PAGE BLANK (uspto)

(19)日本国特許庁 (J P)

(12) 特 許 公 報 (B 2)

(11)特許番号

第2820189号

(45)発行日 平成10年(1998)11月5日

(24)登録日 平成10年(1998)8月28日

(51)Int.Cl.⁸

識別記号

F I

G 0 5 B 19/4155

G 0 5 B 19/403

X

G 0 6 F 9/46

3 4 0

G 0 6 F 9/46

3 4 0 B

請求項の数6 (全 30 頁)

(21)出願番号 特願平5-175490

(22)出願日 平成5年(1993)7月15日

(65)公開番号 特開平7-36529

(43)公開日 平成7年(1995)2月7日

審査請求日 平成8年(1996)12月5日

(73)特許権者 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72)発明者 南角 茂樹

名古屋市東区矢田南五丁目1番14号 三

菱電機株式会社 名古屋製作所内

(74)代理人 弁理士 宮田 金雄 (外2名)

審査官 千葉 成就

(58)調査した分野(Int.Cl.⁸, D B名)

G05B 19/4155

G06F 9/46 340

(54)【発明の名称】 数値制御装置の制御ソフトウェア実行システム

(57)【特許請求の範囲】

【請求項1】 リアルタイムオペレーティングシステム上で制御ソフトウェアを実行したときに起動される複数のタスクの共同作業によって数値制御装置としての機能を実現する数値制御装置のタスクの機能を、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で実行するプロセスまたはタスク中のスレッドに割り当てると共に、スレッド間のスケジューリングの制御用に1つのスレッドを割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システム。

【請求項2】 請求項1の数値制御装置の制御ソフトウェア実行システムにおいて、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステムを用いた計算機

と、その計算機に通信回線を介して接続された数値制御装置とを具備してなり、前記計算機と前記数値制御装置の間に分散型共有メモリの機能を持たせ、前記数値制御装置のタスクの機能を、前記分散型共有メモリの機能を利用して、前記計算機と前記数値制御装置とで分散して処理することを特徴とする数値制御装置の制御ソフトウェア実行システム。

【請求項3】 請求項1または請求項2の数値制御装置の制御ソフトウェア実行システムにおいて、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステムを用いた計算機と、その計算機に通信回線を介して接続された数値制御装置とを具備してなり、前記計算機は前記数値制御装置への割り込み要求の有無を監視することを特徴とする数値制御装置の制御ソフトウェア実行システム。

【請求項4】 請求項1から請求項3のいずれかの数値制御装置の制御ソフトウェア実行システムにおいて、複数のCPUカードを有するCPUプールを具備し、そのCPUカード中のCPUを前記数値制御装置のCPUとして割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システム。

【請求項5】 請求項4の数値制御装置の制御ソフトウェア実行システムにおいて、前記CPUカード中のCPUを前記複数の数値制御装置のCPUとして割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システム。

【請求項6】 請求項5の数値制御装置の制御ソフトウェア実行システムにおいて、前記CPUプール中の1つのCPUカードを割り込み検査専用のCPUカードとして使用することを特徴とする数値制御装置の制御ソフトウェア実行システム。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 この発明は、数値制御装置の制御ソフトウェア実行システムに関し、さらに詳しくは、数値制御装置の制御ソフトウェア（コントロールプログラム）を汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で実行できる数値制御装置の制御ソフトウェア実行システムに関する。

【0002】

【従来の技術】 図27に示す数値制御装置501のハードウェア構成図を用いて従来の数値制御装置の一般的構成を説明する。1は、各部の作動を制御したり、数値制御に必要な演算を行なったりするメインCPUである。2は、各部を結ぶシステムバスである。3は、数値制御装置の主要機能を実現する制御ソフトウェアなどを格納するROM（不揮発性の記憶装置）である。4は、一時記憶やワークエリアなどに用いるRAM（揮発性の記憶装置）である。5は、外部との間でシリアル通信によりデータのやり取りを行なうSIOインタフェース部である。6は、数値制御装置の運転状態を表示したり、数値制御装置に与えた指令を確認したりするための表示装置である。7は、数値制御装置に指令を与えるためのキーボード（入力装置）である。8は、サーボモータを制御するための指令を演算するサーボ制御部である。このサーボ制御部8は、メインCPU1とは別の専用CPUを備えてもよい。9は、サーボ制御部8から受け取った指令を増幅してサーボモータへ駆動信号を出力するサーボアンプである。10は、工作機械（図示せず）の加工部を制御するためのサーボモータである。11は、工作機械との間で、サーボ制御指令以外のデータをやり取りするためのプログラムコントローラ部である。12は、メインCPU1に与えられる、システムクロック（図示せず）と外部割り込み信号を表している。システムクロ

ックは、数値制御装置全体を制御するためのクロック信号である。また、外部割り込み信号は、電源異常や非常停止などのイベント（緊急な出来事）の発生をメインCPU1に通知するための信号である。

【0003】 次に、上記数値制御装置501の動作を説明する。まず、メインCPU1は、ROM3に書き込まれている制御ソフトウェアをシステムバス2を通して1命令ずつ順に読み込んで実行する。図28に、命令読込後の処理手順を示す。加工プログラム入力処理21では、SIOインタフェース部5を通して外部から加工プログラム20を読み込み、RAM4に格納する。そして、加工プログラム20をブロック（所定の単位）ごとに内部データに変換する。補正計算処理22では、ブロックごとの内部データを処理し、増分移動量を算出する。また、工具径や工具長などを補正する。さらに、内部の座標値の更新処理を行なう。設定表示処理23では、数値制御装置の各種データを表示装置6に表示する。また、キーボード7を用いてオペレータが入力した各種設定データをRAM4に格納する。補間処理24では、補正計算処理22の処理結果を用いて、微小時間ごとの各軸の移動量を算出する。サーボ処理25では、補間処理24の処理結果を用いて、さらに小さな単位時間ごとの各軸の移動量に変換する。更に、サーボモータ10からのフィードバック制御（図示せず）を行なう。プログラムコントロール処理26では、工作機械との間での入出力処理や主軸の制御など、工作機械の周辺の制御などを行なう。

【0004】 さて、上記の数値制御装置501は、図27を用いて先に説明したようにメインCPU1に外部割り込み信号12を入力して割り込み処理を行なわせることで、非常停止などの緊急事態に対処することが出来る。メインCPU1は、外部割り込み信号12（図27）が入力されると、予め指定された別処理を実行し、それら別処理が終了した後に、通常の命令に復帰する。

【0005】 図29は、割り込み処理の概念図である。30～36は、通常の命令である。37～40は、割り込み命令である。41は、通常の命令への復帰命令である。例えば、メインCPU1が通常の命令33を実行中に外部割り込み信号12（図27）が入力されると、メインCPU1は、通常の命令33の処理終了後、割り込みを検出し、予め指定されていた割り込み命令37の実行を開始する。そして、割り込み命令37～40の実行を終了した後、復帰命令41を実行して、通常の命令34に復帰し、通常の命令35、36を実行する。なお、割り込み処理の処理時間が長くなると多重割り込みなどを考慮する必要が出てくるので、割り込み処理の処理時間をできるだけ短くすることが望ましい。

【0006】 ところで、上記の数値制御装置501の制数値制御装置の制御ソフトウェアにより実現すべき機

能の多様性のためソフトウェア量も大きくなり、多人数数値制御装置の制御ソフトウェアの機能ごとに、処理に必要な応答時間（ターンアラウンドタイム、デッドライン）が異なる。例えば、サーボ処理25（図28）は、処理結果の算出が遅れると切削が止って被加工物が不良品になってしまうので、実時間処理でなくてはならない。一方、表示装置6への表示処理などは、多少遅れても不都合は生じないので、実時間処理でなくてもメインCPU1が実行すべき割り込み処理に、多くの種類がある。

これらの理由により、数値制御装置501の制御ソフトウェアは、一般にリアルタイムオペレーティングシステムの制御のもとで実行される機能ごとのタスクを実行単位とすることが多い。

【0007】次に、リアルタイムオペレーティングシステムにより各タスクを制御する方法を説明する。あるタスクを定期的に行わせたり、あるタスクを実行する時間を制限したりするために、通常はメインCPU1（図27）に対してある一定周期で割り込み（図示せず）を入れる。（これをシステムクロックと呼ぶ）リアルタイムオペレーティングシステムは、システムクロック割り込みがある度に各タスクの状態を調べ、実行中のタスクを止めて別のタスクを実行させたりする。（これをスケジューリング、またはディスパッチと呼ぶ）また、リアルタイムオペレーティングシステムにおいては、各タスクに優先順位（実行の優先度）づけを行なう。この優先順位の意味は、より低い優先順位のタスクを実行中に、より高い優先順位のタスクを実行する準備ができたときは、その優先順位の低いタスクの実行を中断させて優先順位の高いタスクを実行させることをいう。（これを横取りという）

【0008】図30は、各タスク動作の時間的関係を示すタイムチャートである。縦軸は各処理の実行状況（実行または停止）を表し、横軸は時間の経過を表す。なお、図面作成の都合上、各処理としては、リアルタイムオペレーティングシステムと、割り込み処理と、（優先順位が高い順に）サーボ処理タスク、補正計算処理タスク、表示設定処理タスクを用いるものとする。P1は、リアルタイムオペレーティングシステムが動作している最中に割り込みが発生して制御が割り込み処理に移ったことを示す。P2は、割り込み処理が終了した後、リアルタイムオペレーティングシステムのスケジューラに制御が返る様子を示す。なお、スケジューラは、リアルタイムオペレーティングシステムでスケジューリングを行なう部分である。P3は、リアルタイムオペレーティングシステムのスケジューラにより、次に実行するタスクとしてサーボ処理タスクが選ばれたことを示す。P4は、サーボ処理タスクが終了した後、リアルタイムオペ

レーティングシステムに制御が返る様子を示す。P5は、リアルタイムオペレーティングシステムのスケジューリングによって、次に実行するタスクとして補正計算処理タスクが選ばれたことを示す。P6は、補正計算処理タスクが終了した後、リアルタイムオペレーティングシステムに制御が返る様子を示す。P7は、より優先順位の高いタスクを実行する準備ができていないので、優先順位が最も低い表示処理設定処理タスクに移ったことを示す。P8は、割り込みが発生したことで表示処理設定処理タスクを中断し、割り込み処理に制御が移ったことを示す。P9は、割り込み処理を終了した後、リアルタイムオペレーティングシステムのスケジューラに制御が返る様子を示す。（一般に、割り込み処理を終了した後は、もとのタスクに制御は戻らず、オペレーティングシステムのスケジューラに制御が返る。）P10は、再び表示設定処理タスクに制御が移ったことを示す。P11は、システムクロック（図示せず）によってリアルタイムオペレーティングシステムのスケジューラに制御が返ったことを示す。P12は、サーボ処理タスクを実行する準備ができたので、サーボ処理タスクに移ったことを示す。（これは、表示設定タスクがサーボ処理タスクに実行権を横取りされた例である）P13は、サーボ処理タスクはその実行を終了した後、自主的に実行権を放棄して、リアルタイムオペレーティングシステムのスケジューラに制御を返したことを示す。P14は、再び表示設定処理タスクに制御が移ったことを示す。以降、各タスクは同様の動作を繰り返して、処理を続ける。

【0009】なお、図30に示したように、システムクロック以外の割り込みは不定期に発生するため、数値制御装置側ではその発生を予想できない。そこで、各タスクの特性に応じて、処理時間の配分を行なう必要がある。例えば、サーボ処理タスクは、前もって所定の単位時間のサーボモータの一回分の移動量を計算しておくために、所定の単位時間以内に次の移動量を計算しておく必要がある。つまり、少なくとも所定の単位時間以内には繰り返して実行する必要がある。一方、補正計算処理タスクは、例えば1回の実行で、サーボ処理タスクを3回だけ実行するために必要なデータを作成できるので、単位時間3回につき1回だけ実行すれば良い。また、表示設定処理タスクは他の処理がない時だけ実行すれば足る。

【0010】以上で述べたように、数値制御装置501は、リアルタイムオペレーティングシステム上で制御ソフトウェアを実行したときに起動される複数のタスクの共同作業によって数値制御装置としての機能を実現する。

【0011】さて、現在、数値制御装置の制御ソフトウェアの開発マシンとしては、ワークステーションが主流である。そこで、図31に、ワークステーションを用いた数値制御装置の制御ソフトウェア開発のシステム構成

を例示する。100は、コンパイラ、数値制御装置の制御ソフトウェアのソースコードやオブジェクトコードなど、各種ファイルを格納しているハードディスクである。101は、各種ファイルを一括管理したり、クライアントワークステーションの要求に応じてファイルを転送したりするファイルサーバである。102は、他のネットワークと通信するためのゲートウェイマシンである。103は、各種I/Oデバイスを制御するためのI/Oサーバである。104は、他のシステムに繋がっているネットワークである。105は、システム内を結ぶローカルネットワークである。106~108は、クライアントワークステーションである。

【0012】次に、図32に示す流れ図を用いて、数値制御装置の制御ソフトウェアを開発する手順を説明する。まず、ステップ111では、ファイルサーバ101からテキストファイルの編集を行なうエディタをローカルネットワーク105を通じてクライアントワークステーション106~108へダウンロードする。そして、エディタを使ってソースコードを作成する。一般に、複数の人間がそれぞれクライアントワークステーション106~108(図31)を用いて、機能ごとにソースコードを作成する。

【0013】ステップ112では、ファイルサーバ101からコンパイルを行なうコンパイラをローカルネットワーク105を通じてクライアントワークステーション106~108へダウンロードする。そして、コンパイラを使ってソースコードをコンパイルして、数値制御装置で実行可能なオブジェクトコードを作成する。なお、一般に、数値制御装置のメインCPUとクライアントワークステーション106~108のCPUとは異なるので、コンパイラとしては、クロスコンパイラを用いる。

【0014】ステップ113では、コンパイルエラー(コンパイルしたときのエラー)が発生したか否か判定する。コンパイルエラーが発生したらステップ114に進み、コンパイルエラーが発生しなければステップ115に進む。ステップ114では、ソースコードを修正して、ステップ112に戻る。

【0015】ステップ115では、ファイルサーバ101からリンクを行なうためのリンカとロケータを行なうためのロケータをローカルネットワーク105を通じてクライアントワークステーション106~108へダウンロードする。そして、リンカとロケータを用いてリンクとロケータを行ない、ロードモジュールを作成する。リンクは、複数のオブジェクトを1つの機能にまとめ、さらにそれらを結びつけて、モジュールを生成する作業である。ロケータは、モジュールをロケータする数値制御装置のメモリのアドレスを決定する作業である。ステップ116では、ロードモジュールを、他の媒体へコピーする。媒体としては、フロッピーディスク(数値制御装置がフロッピーディスクドライブを備えているとき)

や、ROMがある。

【0016】ステップ118では、数値制御装置は、媒体からロードモジュールを読み出して、メモリへロードする。ステップ118では、数値制御装置上でロードモジュールを実行し、意図した通りに動作するか否かを確認する。意図した通りに動作すれば処理を終了し、意図した通りに動作しなければ、ステップ114に戻る。ステップ118、119、114の一連の処理をデバッグと呼ぶ。

【0017】ここで、ワークステーションおよびワークステーションのオペレーティングシステムについて説明する。ワークステーションには、以下に示すような特徴ワークステーションにおいては多くのユーザが同時に使用して、その各ユーザに対して公平なオペレーティングシステムのサービスが要求される。つまり、ユーザによる優先順位のようなものも普通は存在しないし、また必ず決まった時間以内に処理しなくてはならない処理は、ほとんど無い。

そのため、ワークステーションのオペレーティングシステムとしては、多数の利用者が端末と対話しながら処理を行えるタイムシェアリングオペレーティングシステムを用いることが多い。

【0018】タイムシェアリングオペレーティングシステムにおいて、あるまとまった処理の単位はプロセスと呼ばれる。プロセスは、リアルタイムオペレーティングシステムにおけるタスクと類似している。但し、一般に、リアルタイムオペレーティングシステムのもとでの各タスクのアドレス空間は1つの同じものである(線形アドレス空間と呼ぶ)のに対して、タイムシェアリングオペレーティングシステムでは、プロセスごとに固有なアドレス空間を持っている。具体例で示せば、リアルタイムオペレーティングシステムでは、CPUがアクセスして出来るアドレス(これを論理アドレス空間と呼ぶ)が0番地から100000番地までであったと想定すると、リアルタイムオペレーティングシステムが0番地から10000番地を占め、タスク1が10001番地から20000番地を占め、タスク2が20001番地から28000番地までを占める。タスク1の20000番地もタスク2の20000番地も同じところを指し、同じデータを示す。そこで、タスク間でアドレスをやり取りすることで、お互いのデータなどを参照できる。一方、タイムシェアリングオペレーティングシステムでは、プロセスごとに固有なアドレス空間を持っているので、プロセス1とプロセス2がどちらもアドレス0番地からアドレス100000番地までを持っていると想定すると、プロセス1の20000番地とプロセス2の20000番地では実際のメモリ上の異なるところを指すので、そのデータも異なる。

【0019】次に、プロセスごとに固有なアドレス空間の実現方法を説明する。図33は、クライアントワークステーションのメモリ態様図である。130はクライアントワークステーション、131はCPU、132はアドレス変換機構、133は変換テーブル、134は主記憶（メモリ）、135はネットワーク、136はファイルサーバ、137はハードディスクである。

【0020】さて、クライアントワークステーション130において、CPU131で区別できるアドレス空間（論理アドレス空間）は、一般に、主記憶につけられた物理的な番地（物理アドレス空間）よりも大きい。そこで、仮想記憶の手法により、物理アドレス空間外の論理アドレスをハードディスクなどの2次記憶装置に割り当てることで、2次記憶装置をも主記憶とみなせるようになる。つまり、主記憶の大きさが実装容量に制限されなくなるので、アドレス空間の制限を気にせずにソフトウェアを作成できるようになる。

【0021】次に、仮想記憶の手法をさらに詳しく説明する。図33において、CPU131はアドレス変換機構132に対して論理アドレスを送る。アドレス変換機構132では、与えられた論理アドレスの一部または全部を取り出して変換テーブル133の配列の何番目かを示すインデックスとして用いる。変換テーブル133の内容が主記憶134のアドレスを示していれば主記憶134に対して所定の動作を行なう。また、変換テーブル133の内容がハードディスク137のブロック番号などを示していればファイルサーバ136に対してブロック番号に対応した内容を要求し、ネットワーク135を介して内容を受け取る。主記憶134に未使用領域があれば、受け取ったデータを未使用領域に格納し、データを格納した領域を示すように変換テーブル133の内容を書き換える。また、主記憶134に未使用領域が存在しなければ、主記憶134に未使用領域が無ければ、主記憶134の既使用領域を選び、その内容をハードディスク137に書き戻し、その領域を新たに使用する。なお、主記憶134から変換テーブル133を逆にたどるソフトウェアの機能も備えられている。（これをコアマップと呼ぶ）

【0022】さらに、プロセスごとに変換テーブルを持つことにより、各プロセスが同じ論理アドレス空間を利用することが出来る。この様子を図34に示す。140はプロセス1、141はプロセス2、142はプロセス1の変換テーブル、143はプロセス2の変換テーブル、144は実メモリを示す。いま、プロセス1（140）とプロセス2（141）のいずれも、アドレス0番地（論理アドレスの0番地）をアクセスする場合を想定する。このとき、プロセス1（140）は、変換テーブル142の0番地の部分を指している。また、プロセス2（141）は、変換テーブル143の0番地の部分を指している。ところで、変換テーブル142の0番地と

変換テーブル143の0番地は、実メモリ144上の異なる領域を指している。つまり、各プロセスが論理アドレス空間をフルに利用することが出来る。

【0023】各プロセスは、例えば図35に示すように、論理アドレス空間をいくつかの領域に分けて使用する。155は、1つのプロセスのすべての論理アドレス空間を示す。156は、そのプロセスのテキスト（プログラム）領域を示す。158は、予備領域である。この予備領域は、普段は未使用領域であるが、データ領域やスタック領域が不足したときには、図示の矢印の方向で、データ領域やスタック領域として用いられる。157は、そのプロセスが使用するデータ領域を示す。159は、スタック領域を示す。

【0024】図36は、タイムシェアリングオペレーティングシステムにおけるプロセスモデルを示す。210は計算機、211はプロセス、212は実体（例えばテキストコード）、213は実体において現在実行中のコードを示すプログラムカウンタである。各プロセスは、アドレス空間や、そのプログラムカウンタや、スタックや、レジスタ群などを持っている。なお複数のプロセス間で協調して処理を行なうときには、プロセス間通信の手段を用いて相互に連絡をとる必要がある。

【0025】上記のタイムシェアリングオペレーティングシステムでは、図36に示したように各プロセスが固有のアドレス空間を持つので、プロセスごとの変換テーブルの書き換えなどに手間がかかり、プロセスの切り換えに時間がかかりやすい欠点がある。また、各プロセス内の実行を行なう実体は1個であるので、複雑な処理を行なうアプリケーションプログラムに対応しづらくなってきた。そのため、近年、1つのプロセス中に複数の実体を置き、各実体を共通のアドレス空間のもとであたかも別々のプロセスであるかのように動かせるマルチスレッドオペレーティングシステムが提案されている。

【0026】図37に、マルチスレッドオペレーティングシステムのモデルを示す。210は計算機、211はプロセス、212は実体、213はプログラムカウンタである。実体212aを、スレッド（またはライトウェイトプロセス）と呼ぶ。このスレッドは、プロセスと同様に、固有のプログラムカウンタやスタックを備えており、他のプロセスとは独立して動く。1つのプロセス内の全てのスレッドは同じアドレス空間を持っているので、共通変数を共有することも容易である。つまり、図36を用いて先に説明したタイムシェアリングオペレーティングシステムのプロセスモデルでは、各プロセスが1つずつしかスレッドを持っていないのに対して、マルチスレッドオペレーティングシステムのプロセスモデルでは、1つのプロセスが複数のスレッドを持つことができる。

【0027】一般的に、各プロセスが持っていなければならない情報には次のようなものがある。

- ・アドレス空間
- ・共通変数（大域変数）
- ・開いているファイルの情報
- ・自分が生成したプロセスの情報
- ・タイマー情報
- ・シグナル情報
- ・セマフォの情報

一方、スレッド毎に持っているべき情報は次のようなものがある。

- ・プログラムカウンタ
- ・スタック
- ・レジスタセット
- ・自分が生成したスレッドの情報
- ・自分自身の状態

【0028】1つのプロセス中のすべてのスレッドは、プロセスごとに持つべき上記の情報を共有している。一方、スレッドごとに持つべき上記の情報は、各スレッドが個々に持っている。例えば図38に示すように、1つのプロセスのすべての論理アドレス空間155は、テキスト領域156と、データ領域157と、予備領域158と；スレッド1のスタック領域220と、スレッド2のスタック領域221と、スレッド3のスタック領域222と、スレッド「N-1」のスタック領域223と、スレッドNのスタック領域224とに分れている。つまり、1つのプロセス内の各スレッドが個別に持たねばならない領域のみを別個に割り当てることによって、1つのプロセス内に複数のスレッドを構成できる。

【0029】次に、マルチスレッドオペレーティングシステムにおけるスケジューリングを説明する。図30を用いて先に説明したリアルタイムオペレーティングシステムではタスクごとにスケジューリング行なったのに対して、マルチスレッドオペレーティングシステムでは、スレッドごとにスケジューリングを行なう。但し、マルチCPUのときにはCPUごとにスレッドが割り当てられるので、割り当てられたスレッドごとのスケジューリングを行なえばよい。

【0030】

【発明が解決しようとする課題】上記従来の数値制御装置501では、数値制御装置としての機能を実現するタスクは、数値制御装置のリアルタイムオペレーティングシステムの下でしか走らせられない。そのため、数値制御装置にロードして制御ソフトウェアを実行させなければならず、面倒である。この発明は、上記問題点を解決するためになされたもので、数値制御装置としての機能を実現するタスクを汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で実行できる数値制御装置の制御ソフトウェア実行システムを得ることを目的とする。

【0031】

【課題を解決するための手段】請求項1の発明は、リア

ルタイムオペレーティングシステム上で制御ソフトウェアを実行したときに起動される複数のタスクの共同作業によって数値制御装置としての機能を実現する数値制御装置のタスクの機能を、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で実行するプロセスまたはタスク中のスレッドに割り当てると共に、スレッド間のスケジューリングの制御用に1つのスレッドを割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0032】請求項2の発明は、請求項1の数値制御装置の制御ソフトウェア実行システムにおいて、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステムを用いた計算機と、その計算機に通信回線を介して接続された数値制御装置とを具備してなり、前記計算機と前記数値制御装置の間に分散型共有メモリの機能を持たせ、前記数値制御装置のタスクの機能を、前記分散型共有メモリの機能を利用して、前記計算機と前記数値制御装置とで分散して処理することを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0033】請求項3の発明は、請求項1または請求項2の数値制御装置の制御ソフトウェア実行システムにおいて、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステムを用いた計算機と、その計算機に通信回線を介して接続された数値制御装置とを具備してなり、前記計算機は前記数値制御装置への割り込み要求の有無を監視することを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0034】請求項4の発明は、請求項1から請求項3のいずれかの数値制御装置の制御ソフトウェア実行システムにおいて、複数のCPUカードを有するCPUブールを具備し、そのCPUカード中のCPUを前記数値制御装置のCPUとして割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0035】請求項5の発明は、請求項4の数値制御装置の制御ソフトウェア実行システムにおいて、前記CPUカード中のCPUを前記複数の数値制御装置のCPUとして割り当てることを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0036】請求項6の発明は、請求項5の数値制御装置の制御ソフトウェア実行システムにおいて、前記CPUブール中の1つのCPUカードを割り込み検査専用のCPUカードとして使用することを特徴とする数値制御装置の制御ソフトウェア実行システムを提供する。

【0037】

【作用】請求項1の制御ソフトウェア実行システムでは、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で数値制

御装置の制御ソフトウェアを実行できるようになる。

【0038】そこで、計算機で数値制御装置の制御ソフトウェアの動作を検証しやすくなり、数値制御装置の制御ソフトウェアの信頼性を向上できる。

【0039】また、請求項1の制御ソフトウェア実行システムでは、数値制御装置の制御ソフトウェアのスケジューリングの制御用に1つのスレッドを生成する。これにより、計算機の汎用のマルチスレッドオペレーティングシステムがスレッド間のスケジューリング機能を備えていなくても、汎用のマルチスレッドオペレーティングシステムまたはマルチタスクオペレーティングシステム上で数値制御装置の制御ソフトウェアを実行できるようになる。

【0040】請求項2の制御ソフトウェア実行システムでは、計算機と数値制御装置とでタスクを分散して処理する。これにより、数値制御装置に低性能のCPUを使用でき、低コストにして効率的に数値制御装置の制御ソフトウェアを実行することが出来る。

【0041】また、請求項2の制御ソフトウェア実行システムでは、分散型共有メモリの機能を利用して計算機と数値制御装置とで分散処理する。これにより、1プロセス内の別スレッドを別ノードで分散して走らせることができないマルチスレッドオペレーティングシステムを用いたときにも、計算機と数値制御装置との間でタスクを分散して処理できるので、数値制御装置に低性能のCPUを使用でき、低コストにして効率的に数値制御装置の制御ソフトウェアを実行することが出来る。

【0042】また、本来は数値制御装置が起動すべきタスクを計算機が処理することで、数値制御装置のCPUを不要にできる。

【0043】請求項3の制御ソフトウェア実行システムでは、計算機は数値制御装置への外部割り込み要求の有無を監視する。これにより、数値制御装置のCPUを無くしても割り込み処理に対処でき、数値制御装置の構成を簡略化できる。

【0044】請求項4の制御ソフトウェア実行システムでは、複数のCPUカードを有するCPUプールを備える。これにより、数値制御装置の制御ソフトウェアの処理負荷に応じてCPUを獲得・解放することで処理能力を調整できる。

【0045】請求項5の制御ソフトウェア実行システムでは、CPUプールの中のCPUを複数の数値制御装置のCPUとして割り当てる。これにより、複数の数値制御装置の制御ソフトウェアを同時に実行できる。

【0046】請求項6の制御ソフトウェア実行システムでは、CPUプールの中の1つのCPUカードを割り込み検査専用のCPUとして使う。これにより、外部割り込み信号に迅速に対処することが出来る。

【0047】

【実施例】以下、図に示す実施例によりこの発明をさら

に詳しく説明する。なお、これによりこの発明が限定されるものではない。

【0048】第1実施例。

図1は、この発明の第1実施例の数値制御装置の制御ソフトウェア実行システムの全体構成図である。この制御ソフトウェア実行システムS1は、数値制御装置200a1~200anと、ホスト計算機201と、ハードディスク202と、表示装置203と、キーボード204とを備える。数値制御装置200a1~200anは、リアルタイムオペレーティングシステムのもとで制御ソフトウェアを実行して、数値制御装置としての機能を実現する。この数値制御装置200a1~200anのハードウェア構成は、図27を用いて先に説明した従来の数値制御装置501と同じである。ホスト計算機201は、汎用のマルチスレッドオペレーティングシステムのもとで数値制御装置の制御ソフトウェアを実行する。ハードディスク202は、ホスト計算機201の外部記憶装置である。表示装置203は、ユーザが必要とする各種の情報を表示する。キーボード204は、ユーザからの指示などを受け付ける。外部ネットワーク206は、ホスト計算機201を、汎用のワークステーション（図示せず）などと結ぶ。

【0049】次に、ホスト計算機201により、数値制御装置の制御ソフトウェアを実行する手順を説明する。なお、説明の都合上、数値制御装置の制御ソフトウェアは、予めハードディスク202に格納されているものとする。まず、ユーザは、キーボード204を用いて数値制御装置の制御ソフトウェアを実現するプロセスの起動をホスト計算機201に指示する。ホスト計算機201のマルチスレッドオペレーティングシステムは、数値制御装置の制御ソフトウェアを実現するためのプロセスをハードディスク202から読み出し、プロセス管理テーブルに登録する。

【0050】プロセス管理テーブルは、概念的には、図33と図34を用いて先に説明した変換テーブルと類似している。具体例で説明すれば、プロセス管理テーブルは、変換テーブル133（図33）の配列の1番目からM番目までを、図38に示したテキスト領域156を示すためのスロットとして確保し、変換テーブル133の配列の「M+1」番目からN番目までを、図38に示したデータ領域157を示すためのスロットとして確保したものである。

【0051】次に、ホスト計算機201は、数値制御装置の制御ソフトウェアの実行状況に伴って、図38に示したプロセスの論理アドレス空間155内の各領域を順にハードディスク202から読み出して内部メモリ（図示せず）へローディングする。

【0052】次に、ホスト計算機201は、数値制御装置の制御ソフトウェアを実行したときのタスクに相当するプロセスを生成する。図2に、プロセスを生成するた

めの流れ図を示す。まず、ステップ241では、マルチスレッドオペレーティングシステムに備えられているシステムコールを利用して、数値制御装置としての1機能を実現している1タスク（数値制御装置のリアルタイムオペレーティングシステムのもとでの1タスク）と同じ処理を行なう1スレッドを生成する。具体的には、プロセス管理テーブルに新たなスレッドのスタック領域用のスロットを確保し、マルチスレッドオペレーティングシステムが認識するスレッドの表の中に新たなスレッドを追加する。

【0053】ステップ242では、ステップ241で生成したスレッドに対応した処理を行なう。例えば、数値制御装置のリアルタイムオペレーティングシステムに対するシステムコールを、ホスト計算機201のマルチタスクオペレーティングシステムのスレッドに対するシステムコールに書き換える。なお、変換ライブラリを用いてシステムコールを変換するならば、書き換え負担を軽減できる。

【0054】ステップ243では、数値制御装置のタスクに相当するスレッドをすべて生成したか否か判定する。すべてのスレッドを生成したならばステップ244に進み、生成すべきスレッドが残っていればステップ241に戻る。なお、すべてのスレッドを生成した時点では、ホスト計算機201の内部には、図3の概念図に示すように、マルチスレッドオペレーティングシステム240上に、数値制御装置のタスクと同じ機能を実現するスレッド1（231）～スレッド6（236）を含むプロセス211が構成されることになる。

【0055】ステップ244では、スレッド間のスケジューリングを行なうスケジューリング方式を決定する。例えば、優先順位に応じて各スレッドを走らせるスケジューリング機能をホスト計算機201のマルチスレッドオペレーティングシステムが備えている場合には、システムコールを用いて各スレッドの優先順位を登録する。又は、ホスト計算機201のマルチスレッドオペレーティングシステムは、準備ができた順序でスレッドを走らせるスケジューラを有していることもある。ここで、“走る”とは、そのスレッドを構成するテキスト領域が実メモリ上に存在し、そのテキスト領域をCPUが処理している状態のことである。

【0056】図4は、スケジューラによるスケジューリングの一例を示す概念図である。211は、ホスト計算機201が実行するプロセスである。231～234は、数値制御装置のタスクと同じ機能を実現するスレッドである。240は、マルチスレッドオペレーティングシステムである。250は、準備が出来たスレッドの中から1つのスレッドを選び順に走らせるスケジューラである。251は、他のプロセスである。

【0057】ステップ245では、数値制御装置の制御ソフトウェアが最初に初期化タスクを起動し、その初期

化タスクが他のタスクを起動するような方式であるときは、初期化タスクと同じ処理を行なうスレッドをシステムコールにより起動する。具体的には、マルチスレッドオペレーティングシステムのスケジューリング対象のスレッドの表の中に、初期化タスクと同じ処理を行なうスレッドを加えればよい。これらの処理により、各スレッドは、予め決定された順序に応じて走ることになる。なお、数値制御装置の制御ソフトウェアが上記のような方式でない場合には、従来の数値制御装置の制御ソフトウェアと同じ順序で該当スレッドを起動すればよい。

【0058】上記ではホスト計算機201のマルチスレッドオペレーティングシステムの機能を利用してスレッド間のスケジューリングを行なった。しかし、マルチスレッドオペレーティングシステムがスレッド間のスケジューリング機能を備えていない場合には上記の対応はできない。

【0059】そこで、この場合は、スレッド間のスケジューリングの制御用に1つのスレッドを生成すればよい。

【0060】つまり、ホスト計算機201は、数値制御装置における各タスクをスレッドとするのに加えて、スケジューラ250（図4）と同じ働きをするスケジューラスレッドを生成する。

【0061】図5は、ホスト計算機201により生成されたスケジューラスレッドによるスケジューリングの概念図である。これは、図4を用いて先に説明したスケジューリングの概念に、スケジューラスレッド252を加えたものである。そして、スケジューラスレッド252は、マルチスレッドオペレーティングシステム240上のスケジューラ250から数値制御装置の制御ソフトウェアを実行するプロセス211の実行権を獲得し、スレッド1（231）～スレッド4（234）のスケジューリングをすべて行なう。

【0062】以上の第1実施例の数値制御装置の制御ソフトウェア実行システムによれば、数値制御装置の制御ソフトウェアのスケジューリングの制御用に1つのスレッド（スケジューラスレッド）を生成するので、ホスト計算機の汎用のマルチスレッドオペレーティングシステムがスレッド間のスケジューリング機能を備えていなくても、ホスト計算機における汎用のマルチスレッドオペレーティングシステムのもとで数値制御装置の制御ソフトウェアを実行させることができるようになる。そのため、制御ソフトウェア開発時のデバッグでは、ホスト計算機上で開発された制御ソフトウェアをいちいち数値制御装置にロードして動作を確認する必要がなくなり、開発期間を短縮できる。また、ホスト計算機上に構築されたCAD/CAMシステムに数値制御装置のシミュレータ機能を組み込む際には、数値制御装置の制御ソフトウェアを汎用のマルチスレッドオペレーティングシステム上で動くように組み直さなくてよいので、手間がかから

ない。また、ホスト計算機と数値制御装置のオペレーティングシステム等の違いを考慮したスケジューリングが不要となるので、数値制御装置の既存のソフトウェアを短時間で移植できる。

【0063】なお、上記ではホスト計算機201上で動作する汎用のマルチスレッドオペレーティングシステムのもとで数値制御装置の制御ソフトウェアのタスクに相当するスレッドを走らせたのに対して、数値制御装置に汎用のマルチスレッドオペレーティングシステムを動作させ、そのもとで数値制御装置の制御ソフトウェアのタスクに相当するスレッドを走らせてもよい。つまり、数値制御装置は、上記のようにホスト計算機201が行なったのと同様にして、マルチスレッドオペレーティングシステム上のスレッドとして、数値制御装置の制御ソフトウェアを実行する。

【0064】このようにすれば、数値制御装置の制御ソフトウェアを汎用のマルチスレッドオペレーティングシステム上で動作するソフトウェアとして開発できるので、ホスト計算機で数値制御装置の制御ソフトウェアの動作を検証しやすくなり、数値制御装置の制御ソフトウェアの信頼性を向上できる。

【0065】第2実施例。

この発明の第2実施例の数値制御装置の制御ソフトウェア実行システムは、前記第1実施例の構成に加えて、ホスト計算機と、1つの数値制御装置との間でスレッド間通信を行なわせ且つ分散型共有メモリの機能を利用して、上述のスレッドを分散して走らせるものである。

【0066】図6は、第2実施例の数値制御装置の制御ソフトウェア実行システムS3の全体構成図である。この制御ソフトウェア実行システムS3は、第1実施例の制御ソフトウェア実行システムS1（図1）とほぼ同じ構成である。但し、第1実施例では数値制御装置200a1～200anがホスト計算機201とは孤立していたのに対して、第2実施例では数値制御装置200b1～200bnとホスト計算機201とを内部ネットワーク205を介して接続した点が異なる。

【0067】図7は、数値制御装置200b1のハードウェア構成図である。なお、数値制御装置200b2～200bnも同じ構成である。1は、各部の作動を制御したり、数値制御に必要な演算を行なったりするメインCPUである。2は、各部を結ぶシステムバスである。3は、数値制御装置の主要機能を実現する制御ソフトウェアなどを格納するROM（不揮発性の記憶装置）である。4は、一時記憶やワークエリアなどに用いるRAM（揮発性の記憶装置）である。5は、外部との間でシリアル通信によりデータのやり取りを行なうデータSIOインタフェース部である。6は、数値制御装置の運転状態を表示したり、数値制御装置に与えた指令を確認したりするための表示装置である。7は、数値制御装置に指令を与えるためのキーボード（入力装置）である。8

は、サーボモータを制御するための指令を演算するサーボ制御部である。9は、サーボ制御部8から受け取った指令を増幅してサーボモータへ駆動信号を出力するサーボアンプである。10は、工作機械（図示せず）の加工部を制御するためのサーボモータである。11は、工作機械との間で、サーボ制御指令以外のデータをやり取りするためのプログラムコントローラ部である。12は、メインCPU1に与えられる、システムクロック（図示せず）と外部割り込み信号を表している。システムクロックは、数値制御装置全体を制御するためのクロック信号である。また、外部割り込み信号は、電源異常や非常停止などのイベント（緊急な出来事）の発生をメインCPU1に通知するための信号である。132は、図33を用いて先に説明した仮想記憶を実現するためのアドレス変換機構である。207は、内部ネットワーク205を通じてホスト計算機201や他の数値制御装置200b2～200bnとの間で通信を行なうためのネットワークインタフェースである。

【0068】上記制御ソフトウェア実行システムS3では、ホスト計算機201上に構築された汎用のCAD/CAMシステムによりCAD/CAMデータを生成し、そのCAD/CAMデータを数値制御装置の加工プログラムに変換して、その加工プログラムに基づいて加工を行なわせることが出来る。図8は、ホスト計算機201上に構築された汎用のCAD/CAMシステムの動作の概念図である。ホスト計算機201は、CAD/CAMプロセス340と、データ変換プロセス341と、数値制御装置の制御ソフトウェアを実現する数値制御装置プロセス342とを生成して起動する。そして、CAD/CAMプロセス340にはデータ出力用スレッド343を走らせておく。CAD/CAMプロセス340は通常の動作によりCAD/CAMデータを作成する。データ出力用スレッド343は、CAD/CAMデータをデータ変換プロセス341のスレッド344に入力する。データ変換プロセス341では、内部のスレッドによりCAD/CAMデータを数値制御装置の入力データに変換し、データ出力スレッド345に渡す。データ出力スレッド345は、数値制御装置の入力データを数値制御装置プロセス342のデータ入力（設定）スレッド346に送る。数値制御装置プロセス342は、数値制御装置の入力データを加工プログラムに変換して、内部ネットワーク205を通じて数値制御装置に入力し、加工を行なわせる。

【0069】上記のようにホスト計算機上のCAD/CAMシステムにより生成されたCAD/CAMデータから数値制御装置の加工プログラムを自動作成すれば、加工プログラムを作成するためのユーザ負担を軽減できる。

【0070】次に、スレッドを分散して走らせることについて説明する。

【0071】スレッドの分散の具体例を説明すれば、ある数値制御装置が低性能で安価なメインCPUを有しており、ホスト計算機201が高性能で高価なCPUを有しているときには、簡単な処理を行なうスレッドは数値制御装置に走らせて、複雑な処理を行なうスレッドはホスト計算機201に走らせるのが好適である。又、別の具体例を説明すれば、図28を用いて先に説明した補間処理24やサーボ処理25を行なうスレッドなどは数値制御装置に走らせ、加工プログラム入力処理21や補正計算処理22を行なうスレッドはホスト計算機201に走らせるのが好適である。

【0072】図9は、数値制御装置200b1とホスト計算機201との間で行なわれるスレッド間通信の説明図である。211は、プロセスである。231はスレッド1、232はスレッド2、233はスレッド3、234はスレッド4である。260はノード1、261はノード2である。ここで、ノードとは、ネットワークの枝の端点にあたる情報処理機器の総称である。例えば、ノード1(260)はホスト計算機201、ノード2(261)は数値制御装置200b1である。262は、オペレーティングシステム240aのもとでスレッド1とスレッド2との間の通信を処理するスレッド間通信処理部である。263は、ノード1とノード2とを結ぶノード間ネットワークである。

【0073】いま、ノード1(260)内のスレッド1(231)からノード2(261)内のスレッド3(233)にメッセージを送る場合を想定する。スレッド1(231)はシステムコールによって相手スレッドを指定し、メッセージを送る要求をオペレーティングシステム240aに対して発行する。このシステムコールによって制御は、オペレーティングシステム240aに移る。システムコールの内容を解析することで、制御はスレッド間通信処理部262に移る。スレッド間通信処理部262は、相手スレッドがあるノード(ノード2)を判定し、ノード2(261)に対してノード間ネットワーク263を介してメッセージを送る。そのメッセージは、ノード2(261)のスレッド間通信処理部262に渡り、スレッド間通信処理部262は、指定されたスレッド3(233)に対してメッセージを渡す。

【0074】上記の想定では、スレッド1(231)からスレッド3(233)に直接にメッセージを送るものとしたが、マルチスレッドオペレーティングシステムによってはメールボックスを用いてメッセージを渡すことも出来る。図10は、ノード1(260)内のスレッドがノード2(261)内のスレッド3(233)にメールボックスを用いてメッセージを送る場合の概念図である。スレッド1(231)はシステムコールによって相手スレッドのメールボックスを指定し、メッセージを送る要求をオペレーティングシステム240aに対して発行する。このシステムコールによって、制御はオペレー

ティングシステム240aに移る。システムコールの内容を解析することで、制御はスレッド間通信処理部262に移る。スレッド間通信処理部262は指定されたメールボックスがあるノード(ノード2)を判定して、ノード2(261)に対してノード間ネットワーク263を通してメッセージを送る。そのメッセージはノード2(261)のスレッド間通信処理部262に渡る。スレッド間通信処理部262は、メールボックス264にメッセージを格納する。その後、スレッド間通信処理部262は、スレッド3(233)からポートを指定したメッセージ読み込み要求があったらメッセージを渡す。読み込み要求の方がメッセージの到着よりも速かった場合には、メッセージが到着するまでスレッド3(233)を待機させる。

【0075】次に、分散共有メモリの機能を利用して、スレッドを分散して走らせることについて説明する。

【0076】1プロセス内の別スレッドを別ノードで分散して走らせることができないマルチスレッドオペレーティングシステムでは、分散型共有メモリの機能を利用して、計算機201と数値制御装置200b1とで分散して処理する。すなわち、ホスト計算機201に数値制御装置の制御ソフトウェアの機能を実現するプロセスを走らせ、1つの数値制御装置200b1にホスト計算機201のプロセスとは独立したスレッドを走らせる。その際、1プロセス内のスレッドでのアドレス空間の共有を、分散型共有メモリの機能により実現する。

【0077】図11は、分散型共有メモリの概念図である。ノード1(260)のCPU131が、ある論理アドレスに対応した主記憶134の内容を読むときは、アドレス変換機構132と変換テーブル133を用いて、主記憶134の物理アドレスを算出して、内容を得る。例えば、論理アドレスがノード1(260)の主記憶134に対応しているならば、ノード1(260)の変換テーブル133は主記憶134の物理アドレスを算出する。また、論理アドレスがノード2(261)の主記憶134に対応しているならば、ノード1(260)の変換テーブル133は、ノード間ネットワーク135を介してノード2(261)の変換テーブル133に主記憶134の内容の読出しを要求する。

【0078】上記では、ホスト計算機201と1つの数値制御装置200b1とに分散してスレッドを走らせたのに対して、図12に示すように、ホスト計算機201と複数の数値制御装置200b1~200bnとに分散してスレッドを走らせてもよい。

【0079】例えば、実時間処理が必要な補間処理24やサーボ処理25(図28)を行なうスレッド231は、それぞれの数値制御装置200b1~200bnで走らせ、実時間処理が不要な加工プログラム入力処理21や補正計算処理22(図28)を行なうスレッド232は、ホスト計算機201上で共通に走らせるのが好適

である。なお、ホスト計算機201上のスレッド232は、数値制御装置200b1～200bn対応で少なくとも1つずつ走らせる。

【0080】以上の第2実施例の数値制御装置の制御ソフトウェア実行システムによれば、前記第1実施例の効果に加えて、1プロセス内の別スレッドを別ノードで分散して走らせることができないマルチスレッドオペレーティングシステムを用いたときにも、ホスト計算機201と数値制御装置200bとの間でタスクを分散して走らせるので、数値制御装置には低性能のCPUを使用でき、低コストにして効率的に数値制御装置の制御ソフトウェアを実行することが出来る。

【0081】上記では、キーボードと表示装置を備えた数値制御装置200b1～200bnを用いたが、キーボードと表示装置を備えていない数値制御装置を用いてもよい。図13に示す制御ソフトウェア実行システムS7は、これまでの制御ソフトウェア実行システムと同様の構成であるが、キーボードと表示装置を備えていない数値制御装置200e1～200enを用いた点異なる。

【0082】図14は、数値制御装置200e1のハードウェア構成図である。なお、数値制御装置200e2～200enも同じ構成である。数値制御装置200e1は、第2実施例の数値制御装置200b1（図7相当）から、表示装置6とキーボード7とを取り除いた構成である。

【0083】図15は、この制御ソフトウェア実行システムS7の動作を示す概念図である。数値制御装置200e1の表示設定処理を行なうスレッド273は、他のスレッド272から表示要求があったとき、自分では表示処理を行わず、ホスト計算機201の表示装置を制御するスレッド270に対して内部ネットワーク205を介して表示要求を送る。スレッド270は表示要求を受け取ると、表示装置203に対して表示処理を行なう。つまり、数値制御装置200e1の表示装置に表示する代りに、ホスト計算機201の表示装置203に表示する。（そのときは、ホスト計算機本来の画面を表示できなくなるので、表示装置203はホスト計算機本来の画面または1つの数値制御装置のいずれかの画面を表示することになる）また、ホスト計算機201のキーボード204が数値制御装置200e1への指示を受け付けると、スレッド271は、その指示内容を内部ネットワーク205を介して数値制御装置200e1の表示設定処理を行なうスレッド273へ送る。つまり、数値制御装置200e1のキーボードを用いて入力する代りに、ホスト計算機201のキーボード204を用いて入力する。

【0084】上記によれば、ホスト計算機201の表示装置203に1つの数値制御装置の画面を表示すると共に、ホスト計算機201のキーボード204を用いて1

つの数値制御装置への指示を入力できるようになるので、数値制御装置の表示装置やキーボードがいらなくなり、数値制御装置の構成を簡略化して安価にすることが出来る。

【0085】ホスト計算機201の表示装置としてビットマップディスプレイを用いてもよい。図16は、ビットマップディスプレイを用いた制御ソフトウェア実行システムの全体構成図である。この制御ソフトウェア実行システムS8は、前記制御ソフトウェア実行システムS7とほぼ同じ構成である。但し、数値制御装置200f1～200fnを用いた点と、ホスト計算機201の表示装置203aとしてビットマップディスプレイを用いた点異なる。

【0086】図17は、この制御ソフトウェア実行システムS8における表示処理の概念図である。ホスト計算機201は、ホスト計算機本来の画面である画面1（280）と画面2（281）と画面3（282）に対応して、画面を制御するスレッド284、285、286を生成する。数値制御装置200f1では、表示要求があると、実施例7と同様に、ホスト計算機201のスレッド284、285、286に対して内部ネットワーク205を介して表示要求を送る。そこで、ホスト計算機201は、ホスト計算機本来の画面と数値制御装置200f1の複数の画面を表示装置203aに表示する。

【0087】上記によれば、ホスト計算機の表示装置にホスト計算機本来の画面と1つの数値制御装置の複数の画面を1度に表示するので、ユーザは多彩な情報を1度に得ることができるようになり、各種の指示を入力する際の操作性が向上する。

【0088】ホスト計算機201の表示装置203a（図16相当）にホスト計算機本来の画面と複数の数値制御装置の画面を同時表示できるようにしてもよい。そのためには、図17を用いて先に説明したホスト計算機201のスレッドを、表示対象の数値制御装置200g1～200gnの個数に応じてさらに増やせば良い。

【0089】このようにすれば、ホスト計算機の表示装置に同時表示された画面からホスト計算機本来の情報と複数の数値制御装置の情報を1度に得ることが出来るため、操作性がさらに良くなる。

【0090】マウスなどのポインティングデバイスを用いて任意の数値制御装置に対して指令を与えられるようにしてもよい。図18は、マウスを備えた制御ソフトウェア実行システムS10は、図16とほぼ同じ構成であるが、マウスMなどのポインティングデバイスを用いて任意の数値制御装置200h1～200hnに対して指令を与えられる点異なる。

【0091】図19は、この制御ソフトウェア実行システムS10における表示処理の概念図である。ホスト計算機201は、図17を用いて先に説明した画面を制御

するスレッドに加えて、スレッド292を生成する。このスレッド292は、ユーザにより操作されたマウスMなどのポインティングデバイスの動きを検知して、ポインティングデバイスカーソル291がどの画面にいるかを判断して、それに応じて例えば画面1(280)を制御するスレッド293に対してポインティングデバイスカーソル291を表示するように要求を出す。そのとき、キーボード204(図18)からの入力、画面1(280)への入力とみなして、入力情報を画面1(280)を制御するスレッド293に対して送る。スレッド293は、当該数値制御装置(図示せず)(のスレッド)に対して入力情報を送る。

【0092】上記によれば、ユーザは、指示を与える数値制御装置をホスト計算機の画面上のポインティングデバイスカーソルにより選択できるので、複数の数値制御装置に指示を与える場合でもいちいち移動する必要がなくなる。

【0093】第3実施例.

この発明の第3実施例の数値制御装置の制御ソフトウェア実行システムは、前記第1実施例の構成に加えて、数値制御装置200i1~200in(図示せず)に対する外部割り込み信号をホスト計算機で処理できるようにしたものである。

【0094】図20は、数値制御装置200i1のハードウェア構成図である。なお、数値制御装置200i2~200inも同じ構成である。数値制御装置200i1は、第2実施例の数値制御装置200b1(図7)から、メインCPU1とROM3と表示装置6とキーボード7とアドレス変換機構132を取り除き、割り込み信号処理部300を追加した構成である。この構成は、ホスト計算機201のCPUを数値制御装置200i1~200inでも使用することで、数値制御装置200のメインCPU1およびアドレス変換機構132(図7相当)を不要としたものである。

【0095】次に、この制御ソフトウェア実行システムS11の動作を説明する。なお、説明の都合上、数値制御装置200i1についての動作のみを説明するが、数値制御装置200i2~200inについても同様である。外部割り込み信号12は、数値制御装置200i1に対して電源異常や非常停止などのイベント(緊急な出来事)の処理を要求する時に発生され、数値制御装置200i1の割り込み信号処理部300に伝えられる。このとき、割り込み信号処理部300は、外部割り込み信号12をRAM4などにラッチ(一時的に記憶)しておく。一方、ホスト計算機201は、RAM4などの内容を定期的にチェックするスレッドを生成し、外部割り込み信号12がラッチされていれば、数値制御装置200i1に対して何らかの要求があったと判定して、ホスト計算機201上の数値制御装置の制御ソフトウェアを実現するスレッド(群)に対してそれを知らせる。その後

は、これまでの実施例と同様に動作する。

【0096】以上の第3実施例の制御ソフトウェア実行システムによれば、ホスト計算機が数値制御装置への外部割り込み信号の有無を監視するので、数値制御装置のCPUを無くしても割り込み処理に対処でき、数値制御装置の構成を簡略化できる。

【0097】第4実施例.

図21は、この発明の第4実施例の数値制御装置の制御ソフトウェア実行システムの全体構成図である。この制御ソフトウェア実行システムS12は、ハードディスク202への読み書きを行なうファイルサーバ208と、CPUプール301と、外部ネットワーク206と接続したゲートウェイ部2070を備えている。また、数値制御装置数値制御装置200j1~200jnのハードウェア構成は、200i1~200in(図20)と同じである。

【0098】図22は、CPUプール301の内部構成図である。CPUプール301のスロットには、複数のCPUカード302が挿入されている。各CPUカード302の間は、バス312により結ばれている。これらCPUカード302は、同種のCPUに統一されていなくてもよい。各CPUカード302のCPU311、313~325は、それぞれローカルメモリ310、312~326を備えている。

【0099】次に、この制御ソフトウェア実行システムS12の動作を説明する。CPUプール301の1つのCPUカード302にホスト計算機201のCPUの役目を果たさせる。これをホスト機能CPUカードと呼ぶ。このホスト機能CPUカードが第3実施例と同様の処理により数値制御装置200j1~200jnにCPUを割り当てる。例えば、図22においてCPU311がホスト機能CPUであるとき、数値制御装置200j1のCPUとしてCPU313を割り当てる。このとき、数値制御装置200j1の制御ソフトウェアを実行するスレッドは、ローカルメモリ310とローカルメモリ312(図22)にローディングされて、それぞれホスト機能CPUのマルチスレッドオペレーティングシステムのもとで走る。これらの処理は、第2実施例で述べた処理と類似しているが、スレッド間通信はネットワークではなくバス312を通して行なう点異なる。

【0100】なお、CPUの能力が不足するとき(例えばホスト機能CPUのオペレーティングシステムのもとで走るスレッドの数がある一定量を越えたようなとき)は、CPUプール301(図22)内のCPUカード302に余裕がある限り新たなCPUカード302を獲得することで、CPUの処理能力を調整できる。具体的には、CPUの処理能力を多く必要とする時には複数のCPUカード302を使用し、CPUの処理能力がそれほど必要でない時には1つのCPUカード302を使用すればよい。

CPUカード
p25の
図

【0101】図23は、新たなCPUカード302を獲得するときの動作を示す概念図である。320、321は、ホスト機能CPU311上で走っているスレッドを示している。322～323は、数値制御装置200j1の制御ソフトウェアを実行するCPU313上で走っているスレッドである。

【0102】スレッド322は、CPU312上で走っているスレッドの数を周期的に数える。スレッド320は、スレッド322と周期的にメッセージを交換することでCPU313上で走っているスレッドの数を認識する。スレッドの数が所定値を越え、スレッド320は、数値制御装置200h1の処理を行なうCPU313を獲得したのと同じ手順で新たなCPUカードを獲得する。ここでは、説明の都合上、図22におけるCPU325を新たに獲得するものとする。まず、新たに走らせるべきスレッドのコードと、CPU325のオペレーティングシステムをハードディスク202（図21）から読み出し、ローカルメモリ326にローディングする。初期化スレッドを走らせた後、スレッド322に対して新たに獲得したCPU325についての情報を送る。次に、それまでCPU313で走らせていたスレッド322をCPU325に移して走らせる。

【0103】以上の第4実施例の制御ソフトウェア実行システムによれば、複数のCPUカードを有するCPUプールを備えるので、数値制御装置の制御ソフトウェアの処理負荷に応じてCPUを獲得・解放することで処理能力を調整できる。

【0104】第5実施例。

この発明の第5実施例の数値制御装置の制御ソフトウェア実行システムは、第4実施例とほぼ同じである。但し、第4実施例では1度に1つずつの数値制御装置を制御するのに対して、第5実施例では1度に複数の数値制御装置を制御できる点が異なる。

【0105】この制御ソフトウェア実行システムS13の動作は、第4実施例の制御ソフトウェア実行システムS12の動作とほぼ同じである。但し、CPUプール301（図21）中のホスト機能CPUカード上のスレッドが第4実施例と同様の処理により数値制御装置に順にCPUを割り当てていく。

【0106】以上の第5実施例の制御ソフトウェア実行システムS13によれば、CPUプールの中のCPUを複数の数値制御装置のCPUとして割り当てることができるので、第4実施例による効果に加えて、複数の数値制御装置の制御ソフトウェアを同時に実行できるようになる。

【0107】第6実施例。

この発明の第6実施例の数値制御装置の制御ソフトウェア実行システムS14は、第5実施例とほぼ同じである。但し、第6実施例では、CPUプール301（図22）中の1つのCPUカード302を割り込み検査専用

のCPUカードとして使用する点が異なる。

【0108】この制御ソフトウェア実行システムS14の動作は、第5実施例の制御ソフトウェア実行システムS13の動作とほぼ同じである。但し、CPUカード302のローカルメモリへのコードのローディングはホスト機能CPUのスレッドが行ない、割り込みを発見した場合には、当該数値制御装置を制御しているCPUカード302上のスレッドにメッセージを送ってそれを通知する。また、割り込みが当該数値制御装置に対する最初の割り込みであった場合には、それをホスト機能CPUのスレッドに送って、これまで述べたような手順でその数値制御装置の動作を開始させる。

【0109】以上の第6実施例の制御ソフトウェア実行システムによれば、CPUプールの中の1つのCPUカードを割り込み検査専用のCPUとして使うので、外部割り込み信号に迅速に対処することができるようになる。

【0110】図24に示す制御ソフトウェア実行システムS14'は、制御ソフトウェア実行システムS14とほぼ同じ構成である。但し、CPUカード302のCPUごとのローカルメモリ310、312～326（図22相当）に代えて、CPUカード302のCPUで共有できる共有メモリ330を設けた点と、数値制御装置200k1～200knを用いた点とが異なる。

【0111】図25は、数値制御装置200k1のハードウェア構成図である。なお、数値制御装置200k2～200knは、数値制御装置200k1と同じ構成である。数値制御装置200k1は、実施例11の数値制御装置200i1（図20）からRAM4を取り除いた構成である。

【0112】この制御ソフトウェア実行システムS14'の動作は、制御ソフトウェア実行システムS14の動作とほぼ同じであるが、仮想的な分散型共有メモリを用いる代わりに、実際に同じ物理アドレスでスレッドを走らせる。このときは、あるプロセス内のスレッドを任意のノード上で走らせることができる。

【0113】以上のようにすれば、CPUカードのCPUごとにローカルメモリを備えなくても、外部割り込み信号に迅速に対処することができるようになる。

【0114】変形実施例。

数値制御装置200p1（図示せず）に汎用のマルチスレッドオペレーティングシステム自体とデータ入カスレッドだけを予め走らせておき、データ入カスレッドは、ホスト計算機201のデータ出カスレッドから、ホスト計算機201のハードディスク202内にファイルとして管理されている数値制御装置の制御ソフトウェアのコード部分を通信で受け取って、そのコードを数値制御装置200p1のRAM4に順に格納していく、というようにしてもよい。

【0115】このようにすれば、数値制御装置はホスト

計算機からダウンロードされた制御ソフトウェアを実行できるので、ホスト計算機上で開発された制御ソフトウェアを速やかに実行できるようになる。また、ホスト計算機は、数値制御装置の制御ソフトウェアをファイルとして管理するので、管理効率が向上する。

【0116】ダウンロード対象の数値制御装置を1台に限らず、複数台にしてもよい。

【0117】ダウンロード対象の数値制御装置を複数台とすれば、複数の種類の数値制御装置の制御ソフトウェアをファイルとして一括して管理できるため、管理効率がさらに向上する。

【0118】数値制御装置にフラッシュROMを追加してもよい。

【0119】図26は、フラッシュROMを追加した数値制御装置200q1のハードウェア構成図である。なお、数値制御装置200q2～200qnも同じ構成である。数値制御装置200q1は、第2実施例の数値制御装置200b1（図7）に、フラッシュROM1133を追加した構成である。

【0120】次に、この数値制御装置200q1に制御ソフトウェアをダウンロードするときの動作を説明する。数値制御装置200q1には、汎用のマルチスレッドオペレーティングシステム自体と、データ入力スレッドと、フラッシュROM書き込みスレッドだけを予め走らせておく。そして、データ入力スレッドは、ホスト計算機201のデータ出力スレッドから、ホスト計算機201のハードディスク202内にファイルとして管理されている数値制御装置の制御ソフトウェアのコード部分を通信で受け取って、そのコードを順にフラッシュROM書き込みスレッドに渡す。フラッシュROM書き込みスレッドは、与えられたコードをフラッシュROM1133に書き込んでいく。

【0121】上記によれば、数値制御装置の制御ソフトウェアをフラッシュROMに保持しておけるため、数値制御装置を立ち上げる度に制御ソフトウェアをダウンロードしなくてよい。また、フラッシュROMに書き込まれた内容は1度にまとめて消去できるので、制御ソフトウェアの書き換え時間を短縮できる。

【0122】なお、上記説明では、ホスト計算機201のオペレーティングシステムとして、汎用のマルチスレッドオペレーティングシステムを用いたが、汎用のマルチタスクオペレーティングシステムを用いても同等である。

【0123】

【発明の効果】請求項1の制御ソフトウェア実行システムによれば、数値制御装置の制御ソフトウェアのスケジューリングの制御用に1つのスレッドを生成するので、計算機の汎用のマルチスレッドオペレーティングシステムがスレッド間のスケジューリング機能を備えていなくても、計算機における汎用のマルチスレッドオペレーテ

ィングシステムの下で数値制御装置の制御ソフトウェアを実行させることができるようになる。請求項2の制御ソフトウェア実行システムによれば、1プロセス内の別スレッドを別ノードで分散して走らせることができないマルチスレッドオペレーティングシステムを用いたときにも、計算機と数値制御装置との間でタスクを分散して処理できるので、数値制御装置に低性能のCPUを使用でき、低コストにして効率的に数値制御装置の制御ソフトウェアを実行することが出来る。請求項3の制御ソフトウェア実行システムによれば、計算機が数値制御装置への外部割り込み信号の有無を監視するので、数値制御装置のCPUを無くしても割り込み処理に対処でき、数値制御装置の構成を簡略化できる。請求項4の制御ソフトウェア実行システムによれば、複数のCPUカードを有するCPUプールを備えるので、数値制御装置の制御ソフトウェアの処理負荷に応じてCPUを獲得・解放することで処理能力を調整できる。請求項5の制御ソフトウェア実行システムによれば、CPUプールの中のCPUを複数の数値制御装置のCPUとして割り当てることができるので、複数の数値制御装置の制御ソフトウェアを同時に実行できる。請求項6の制御ソフトウェア実行システムによれば、CPUプールの中の1つのCPUカードを割り込み検査専用のCPUとして使うので、外部割り込み信号に迅速に対処することが出来る。

【図面の簡単な説明】

【図1】この発明の第1実施例の制御ソフトウェア実行システムの全体構成図である。

【図2】プロセスを生成するための流れ図である。

【図3】ホスト計算機の内部に構成されたプロセスの概念図である。

【図4】スケジューラによるスケジューリングの概念図である。

【図5】スケジューラスレッドによるスケジューリングの概念図である。

【図6】この発明の第2実施例の制御ソフトウェア実行システムの全体構成図である。

【図7】数値制御装置200b1のハードウェア構成図である。

【図8】ホスト計算機上に構築された汎用のCAD/CAMシステムの動作の概念図である。

【図9】数値制御装置とホスト計算機の間で行なわれるスレッド間通信の概念図である。

【図10】メールボックスを介したスレッド間通信の概念図である。

【図11】分散型共有メモリの概念図である。

【図12】ホスト計算機と複数の数値制御装置とに分散する例の概念図である。

【図13】キーボードと表示装置を備えていない数値制御装置を用いた制御ソフトウェア実行システムの全体構成図である。

【図14】数値制御装置200e1のハードウェア構成図である。

【図15】キーボードと表示装置を備えていない数値制御装置を用いた制御ソフトウェア実行システムの動作を示す概念図である。

【図16】ビットマップディスプレイを用いた制御ソフトウェア実行システムの全体構成図である。

【図17】ビットマップディスプレイを用いた制御ソフトウェア実行システムにおける表示処理の概念図である。

【図18】マウスを備えた制御ソフトウェア実行システムの全体構成図である。

【図19】マウスを備えた制御ソフトウェア実行システムにおける表示処理の概念図である。

【図20】この発明の第3実施例の数値制御装置200i1のハードウェア構成図である。

【図21】この発明の第4実施例における制御ソフトウェア実行システムの全体構成図である。

【図22】CPUブールの内部構成図である。

【図23】新たなCPUカードを獲得するときの動作の概念図である。

【図24】この発明の変形実施例の制御ソフトウェア実行システムの全体構成図である。

【図25】数値制御装置200k1のハードウェア構成図である。

【図26】数値制御装置200q1のハードウェア構成図である。

【図27】従来の数値制御装置のハードウェア構成図である。

【図28】従来の数値制御装置の処理手順を示す説明図である。

【図29】割り込み処理の概念図である。

【図30】各タスク動作の時間的関係を示すタイムチャートである。

【図31】ワークステーションを利用した制御ソフトウェア開発システムの構成図である。

【図32】制御ソフトウェアの開発手順を示す流れ図である。

【図33】クライアントワークステーションのメモリ態様図である。

【図34】各プロセスが同じ論理アドレス空間を利用できることを示す原理図である。

【図35】1つのプロセスの論理アドレス空間内の領域図である。

【図36】タイムシェアリングオペレーティングシステムにおけるプロセスモデルの例示図である。

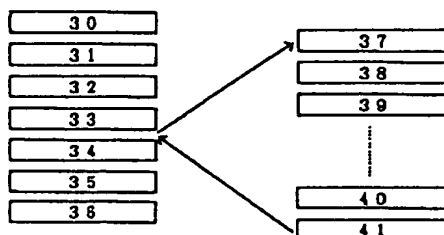
【図37】マルチスレッドオペレーティングシステムにおけるモデルの例示図である。

【図38】1つのプロセスの論理アドレス空間内の別の領域図である。

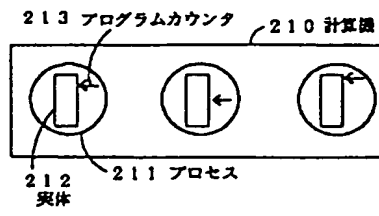
【符号の説明】

- 1 メインCPU
- 2 システムバス
- 3 ROM
- 4 RAM
- 5 SIOインタフェース部
- 6 表示装置
- 7 キーボード
- 8 サーボ制御部
- 9 サーボアンプ
- 10 サーボモータ
- 11 プログラムコントローラ部
- 12 外部割り込み信号
- 132 アドレス変換機構
- 1133 フラッシュROM
- S1～S14 制御ソフトウェア実行システム
- 200a1, ..., 200q1 数値制御装置
- 200kn, ..., 200qn 数値制御装置
- 201 ホスト計算機
- 202 ハードディスク
- 203, 203a 表示装置
- 204 キーボード
- 207 ネットワークインタフェース
- 205, 208 内部ネットワーク
- 206 外部ネットワーク
- 207 ネットワークインタフェース
- 301 CPUブール
- 302 CPUカード
- 2070 ゲートウェイ部
- M マウス

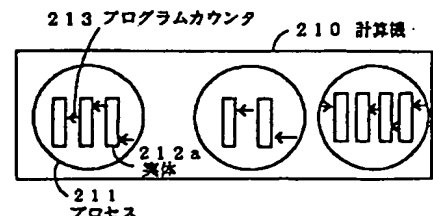
【図29】



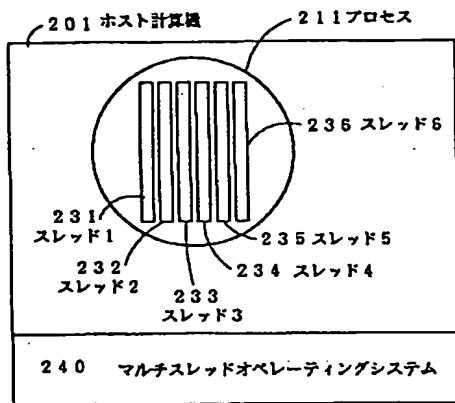
【図36】



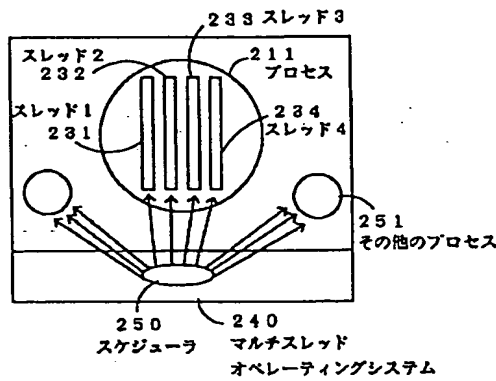
【図37】



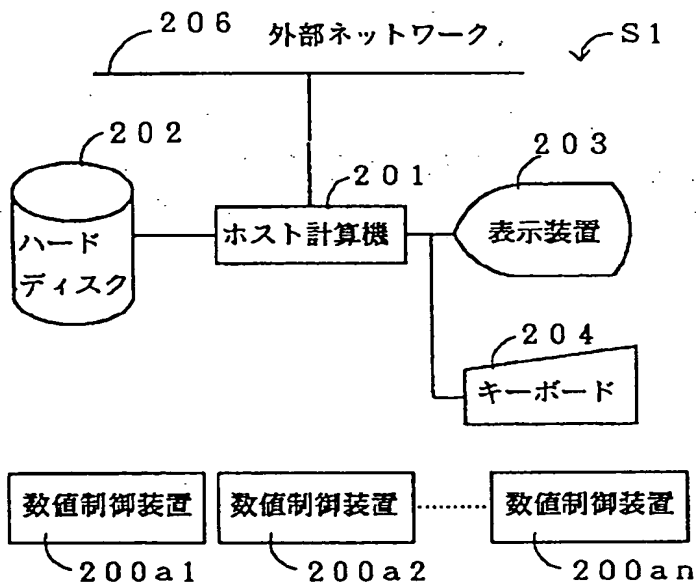
【図3】



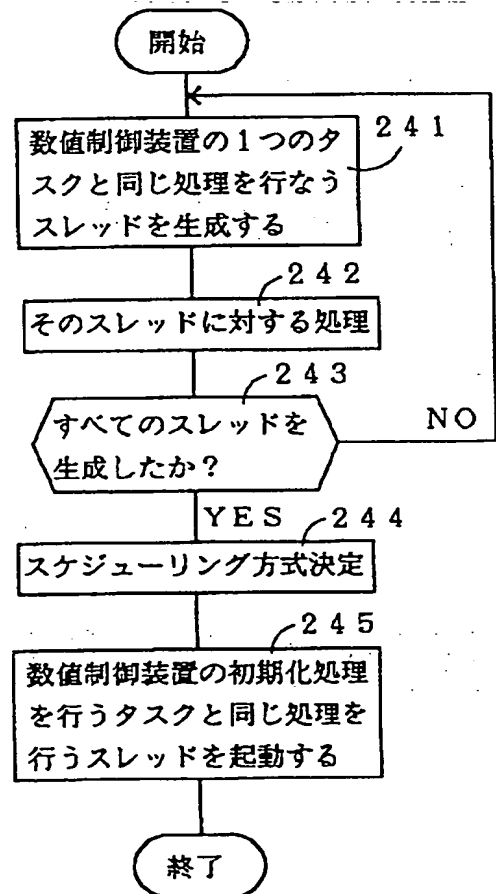
【図4】



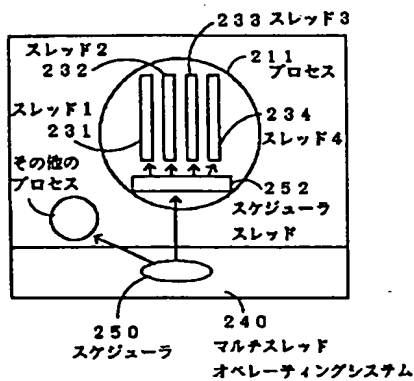
【図1】



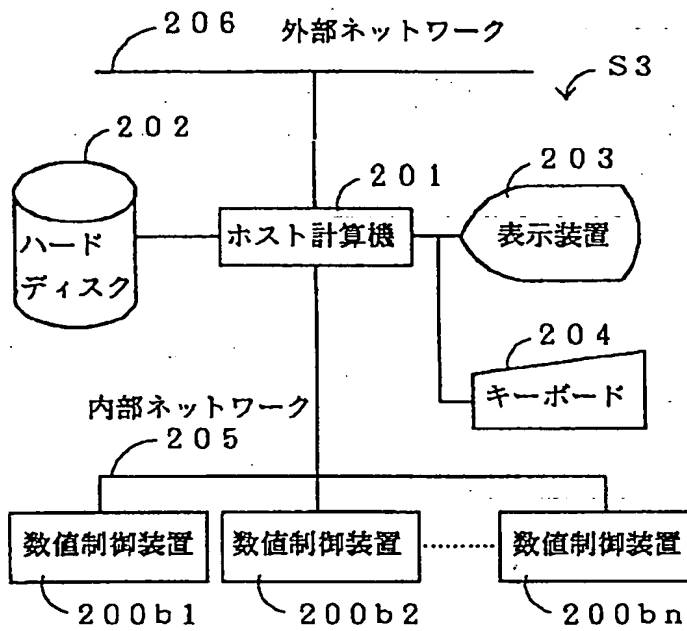
【図2】



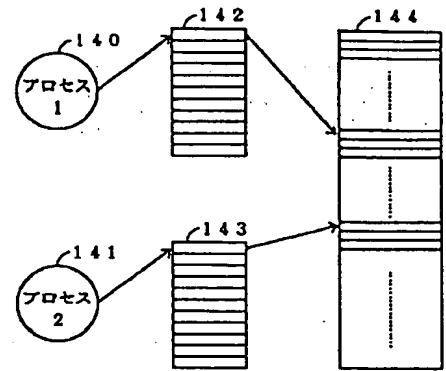
【図5】



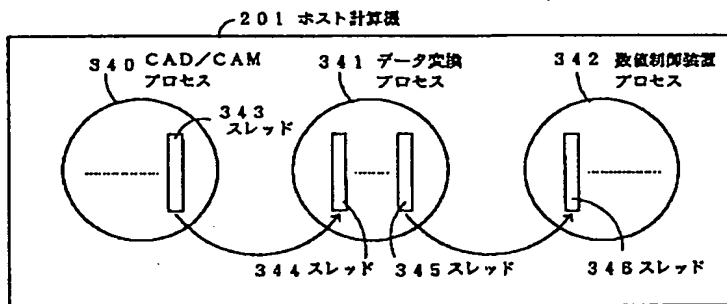
【図6】



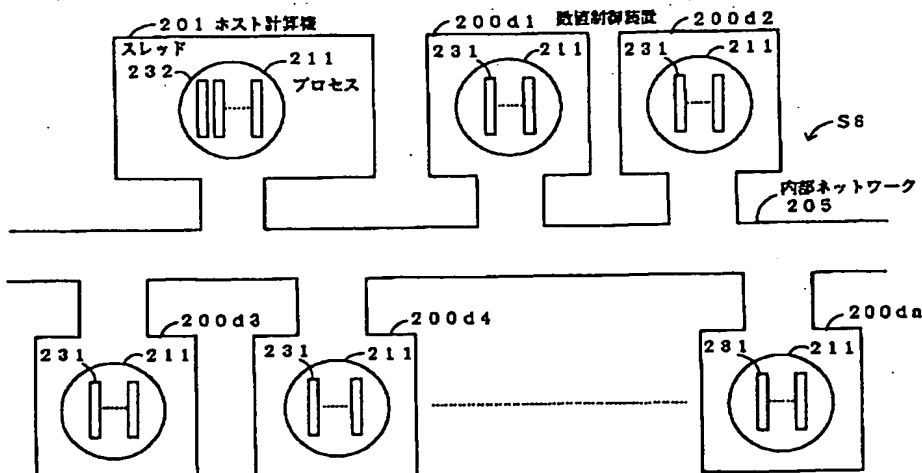
【図34】



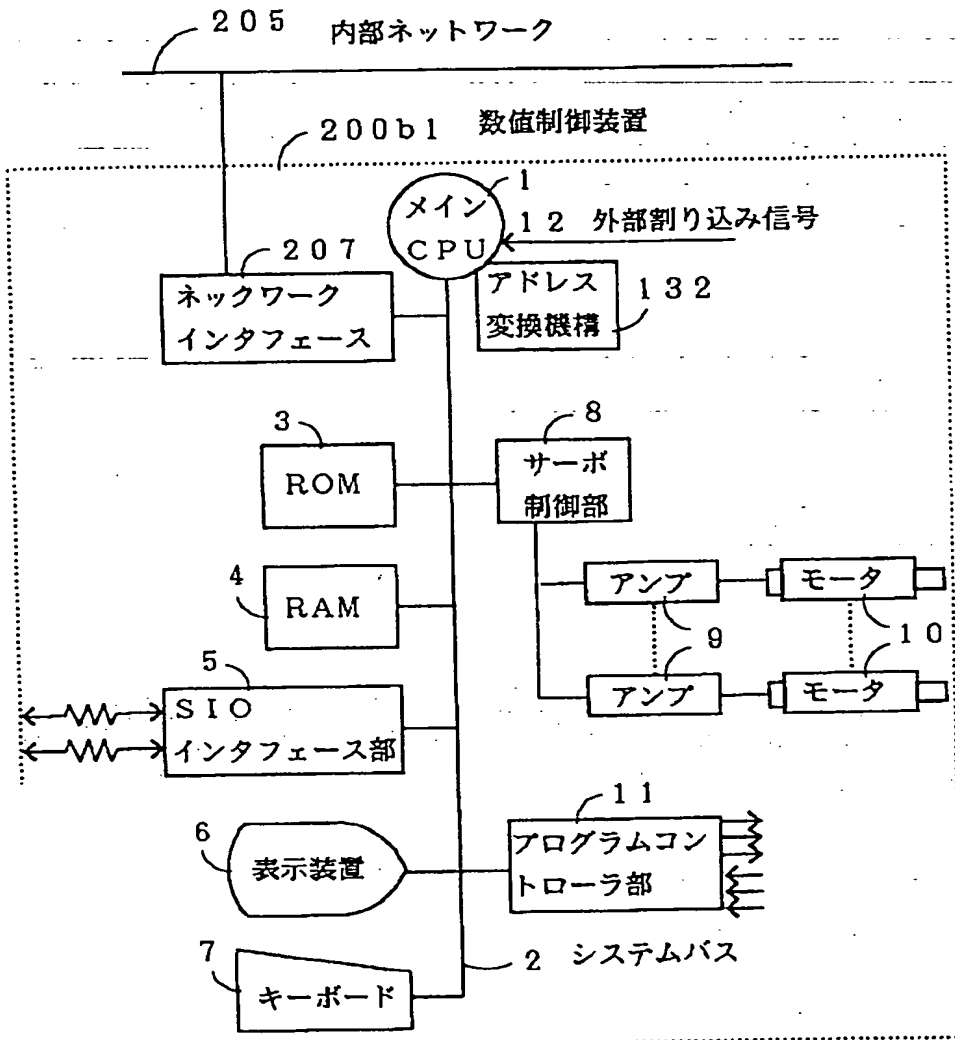
【図8】



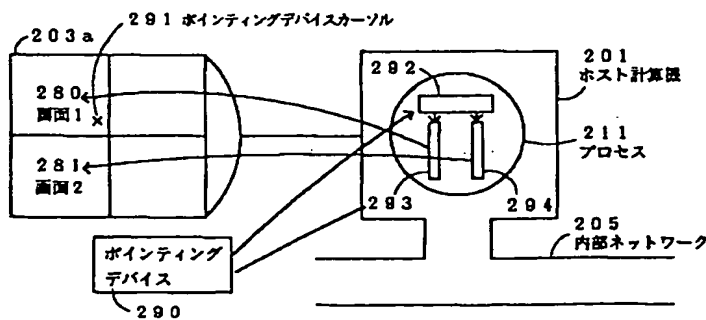
【図12】



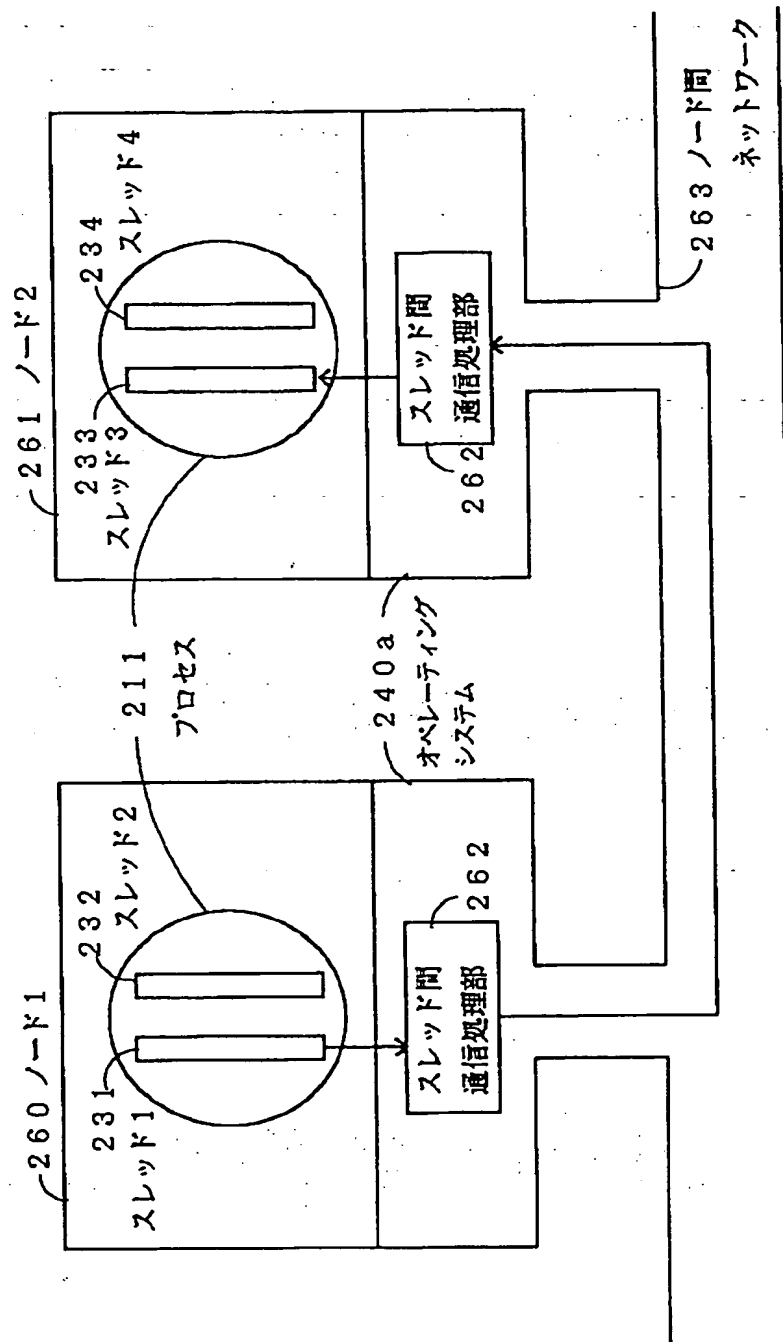
【図7】



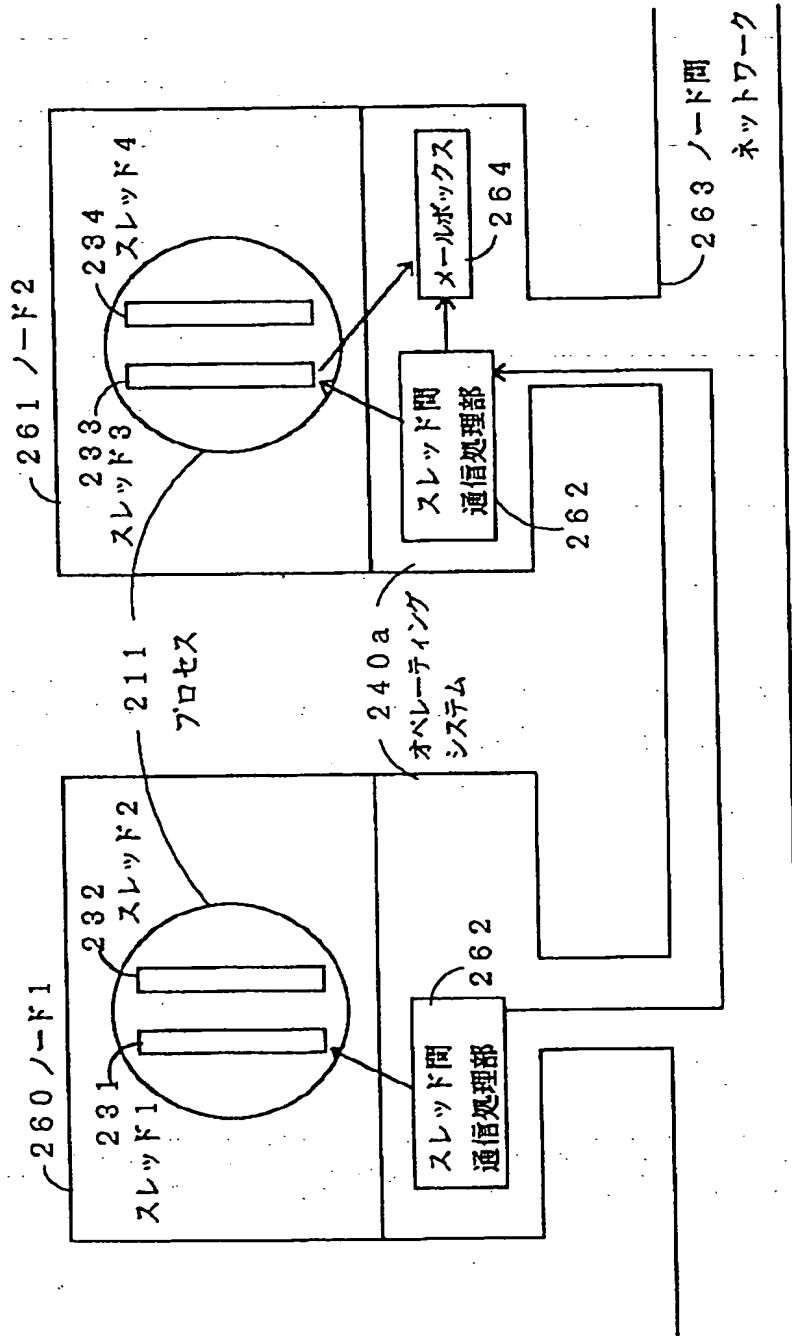
【図19】



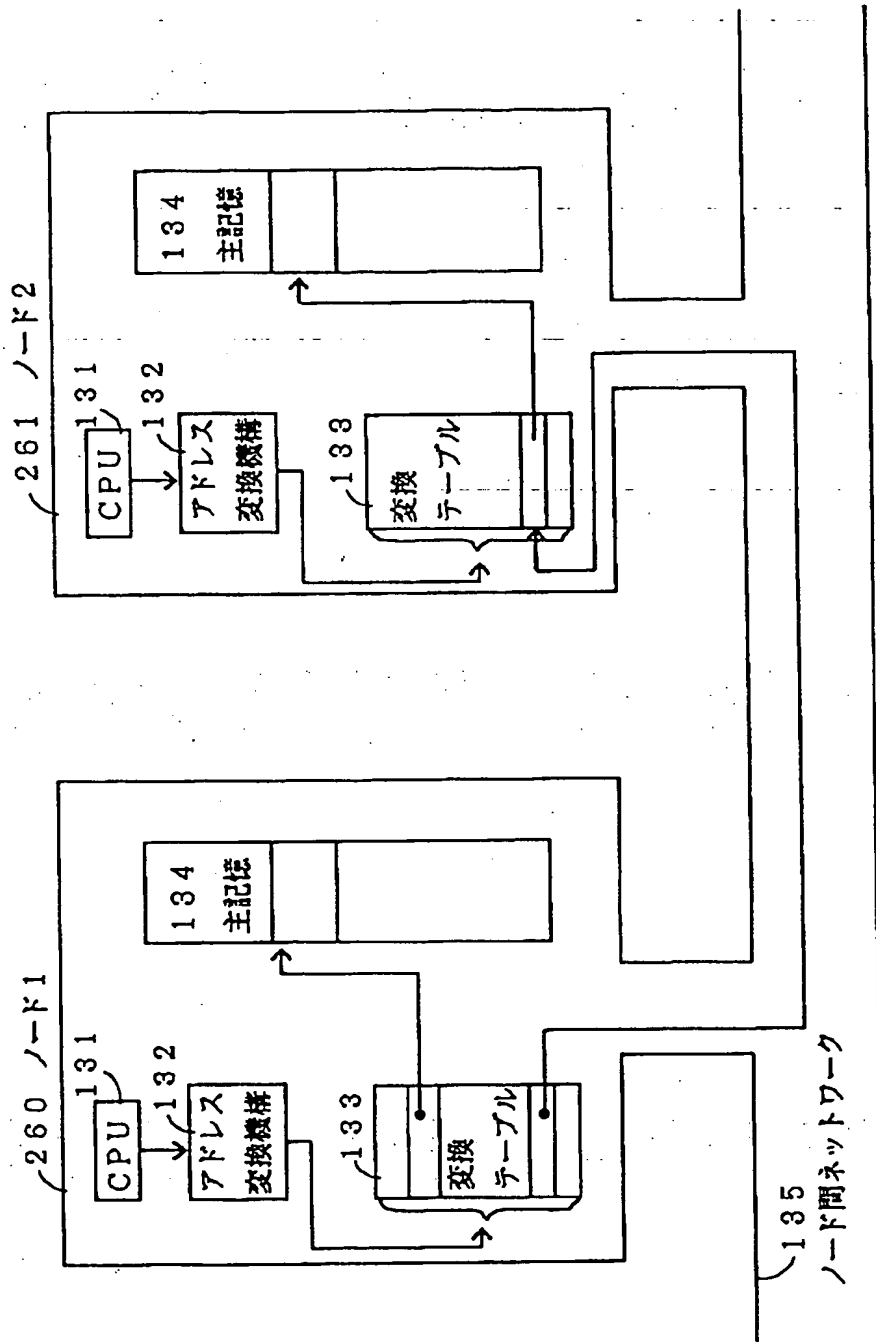
【図9】



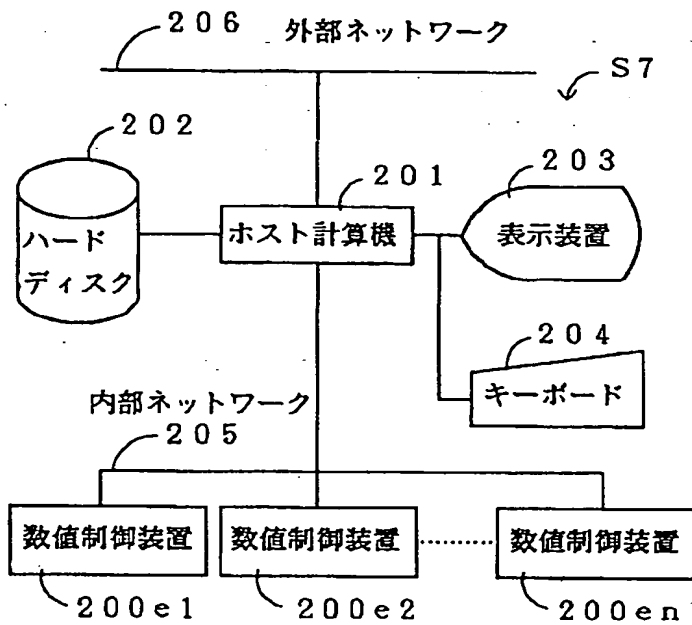
【図1.0】



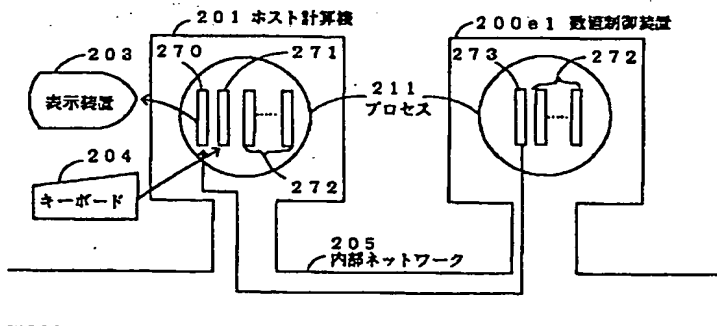
【図11】



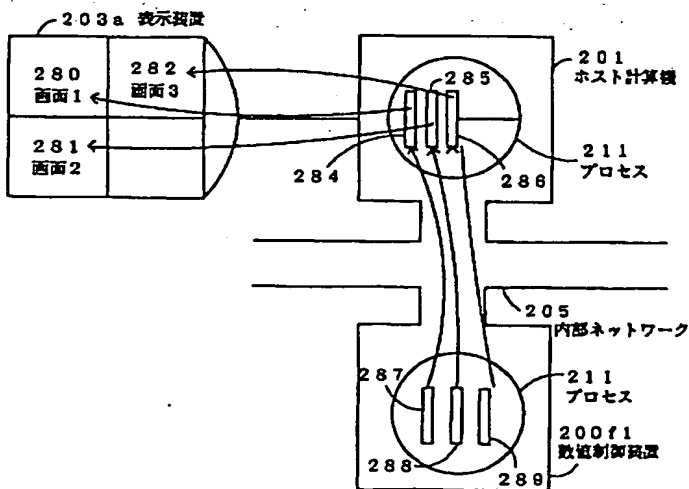
【図13】



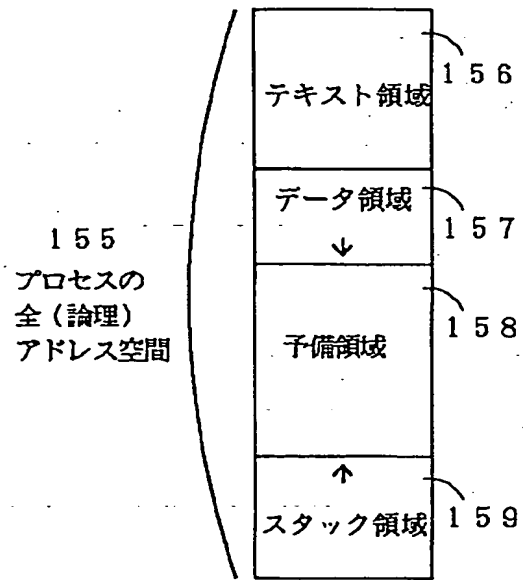
【図15】



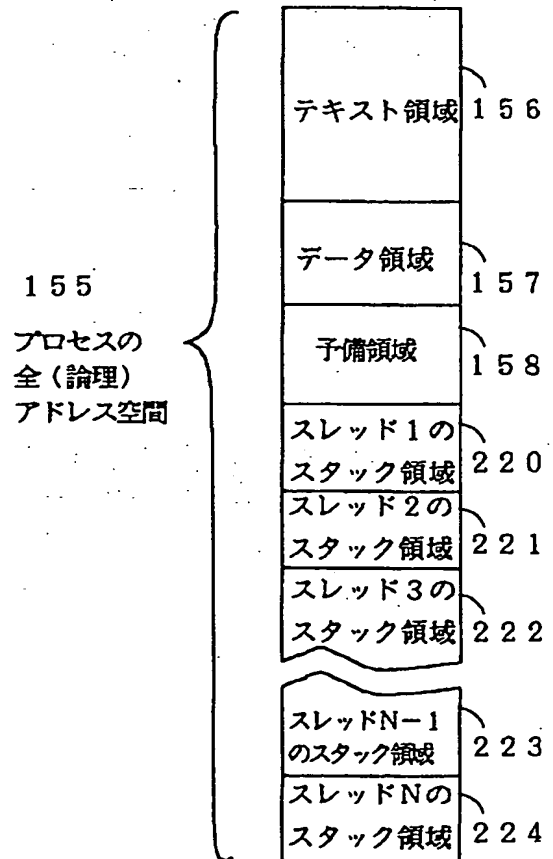
【図17】



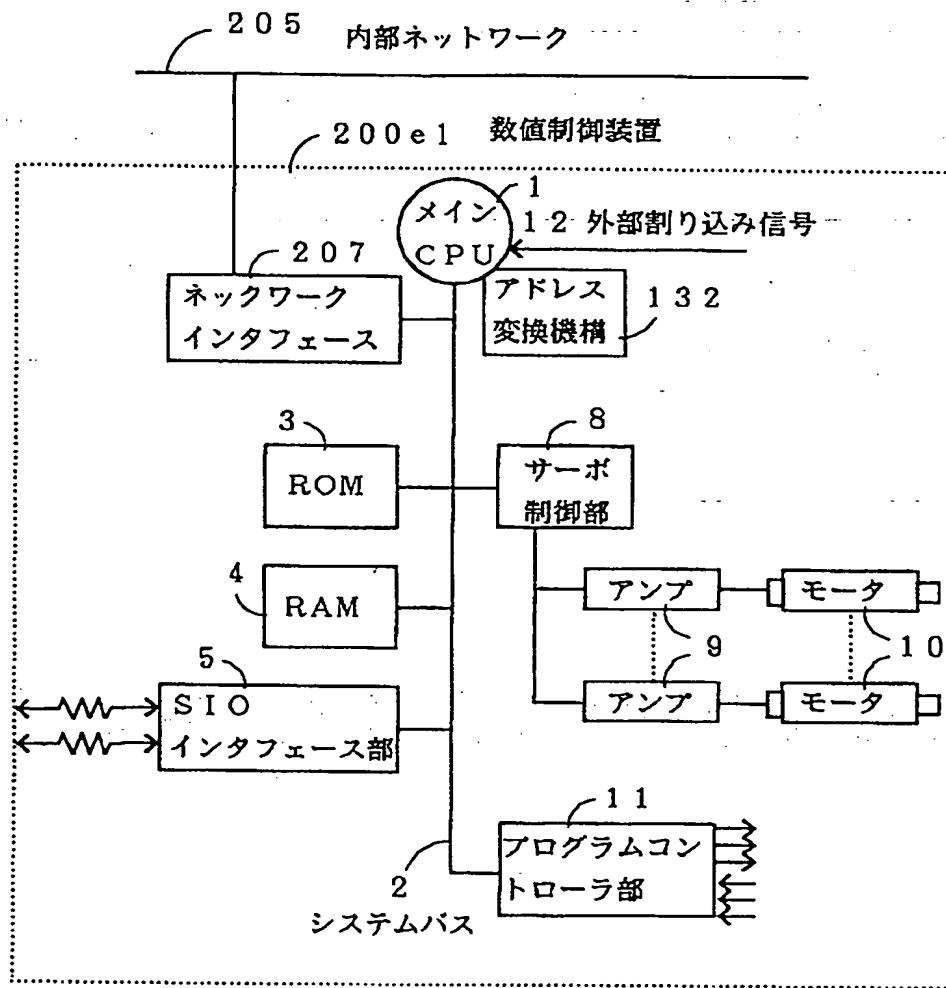
【図35】



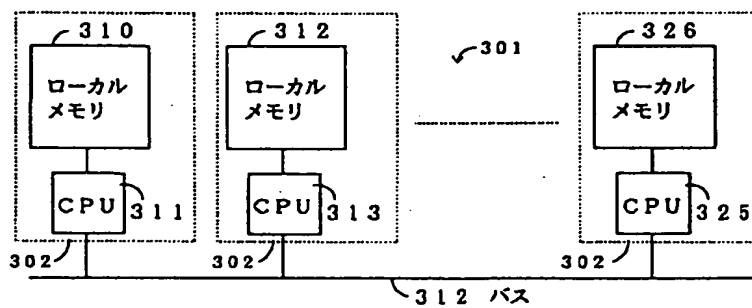
【図38】



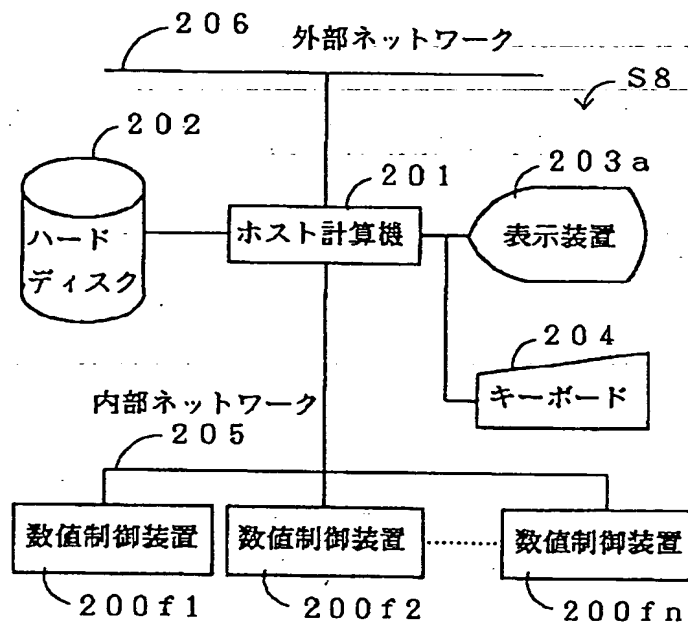
【図14】



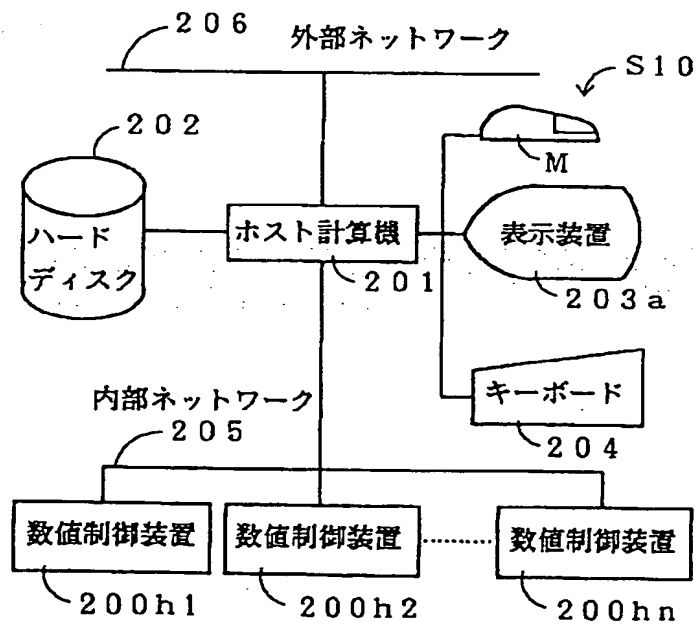
【図22】



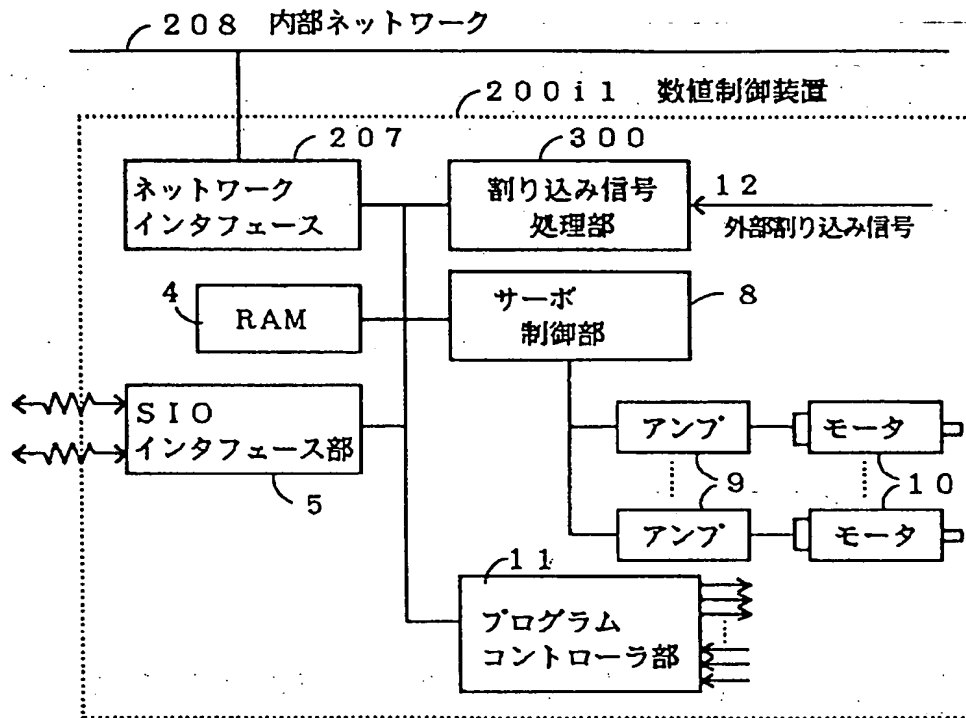
【図16】



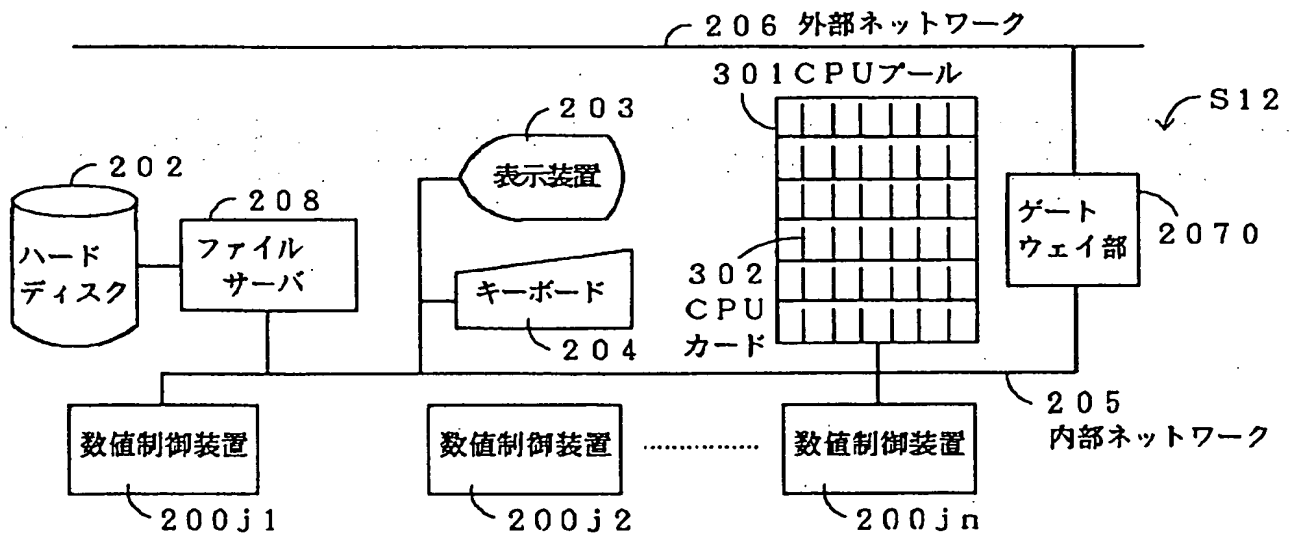
【図18】



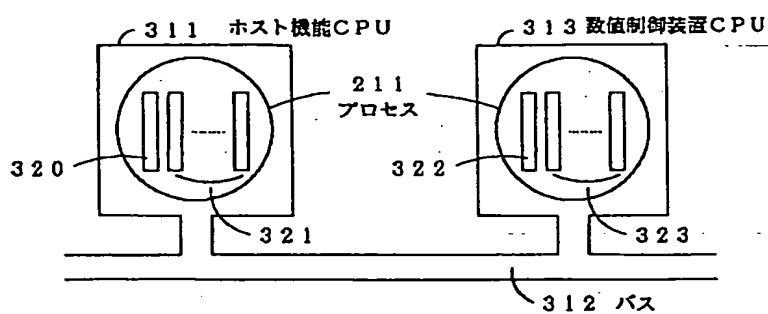
【図20】



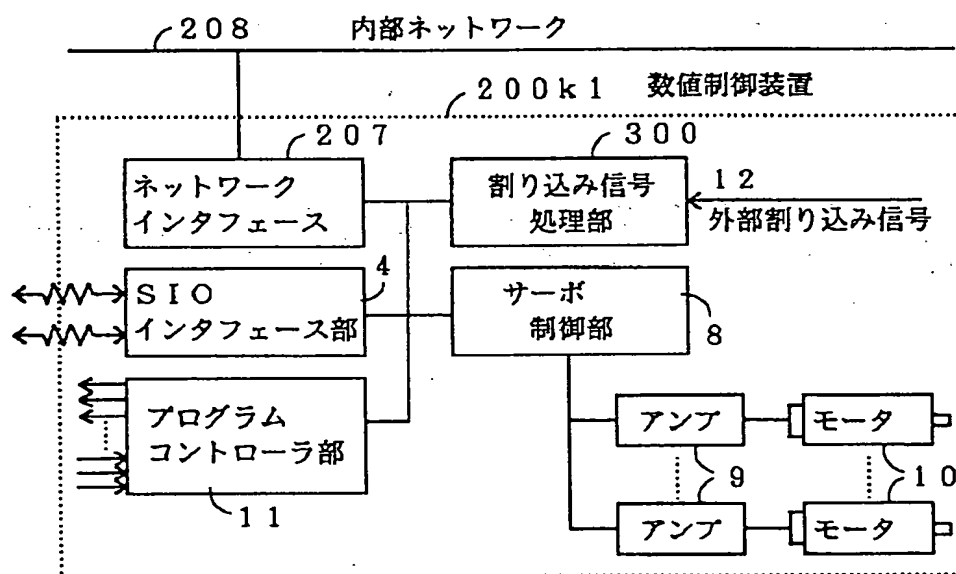
【図21】



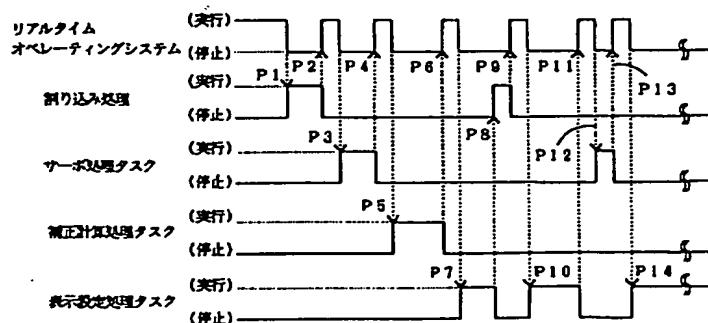
【図23】



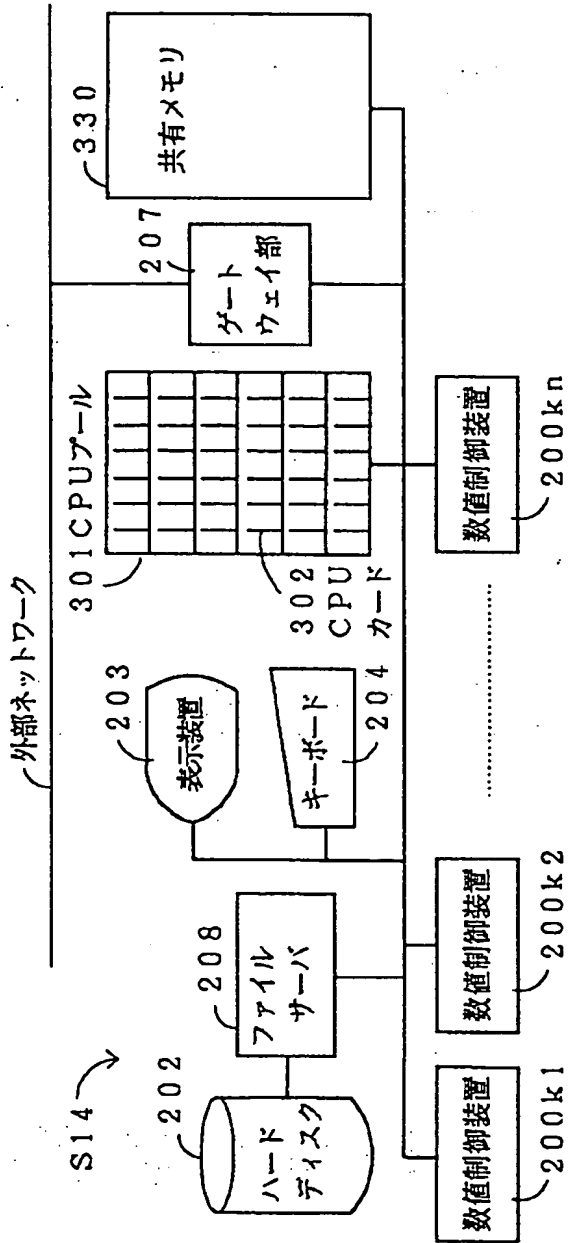
【図25】



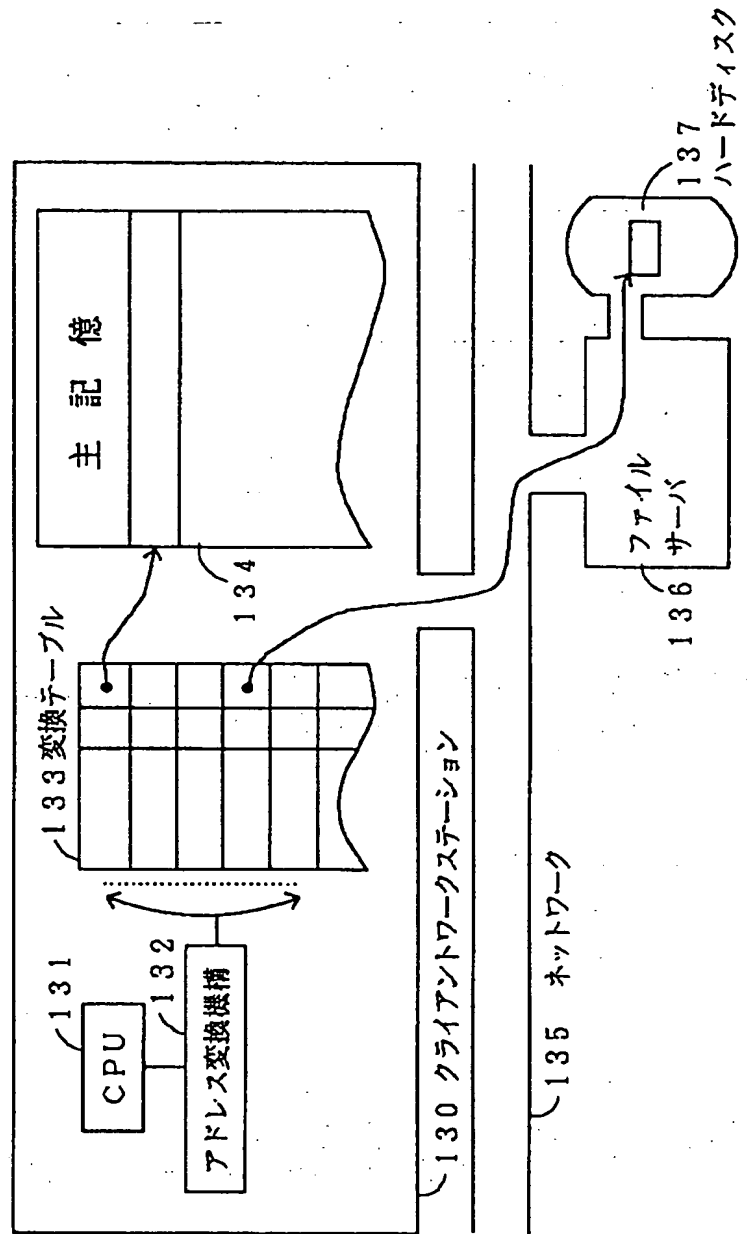
【図30】



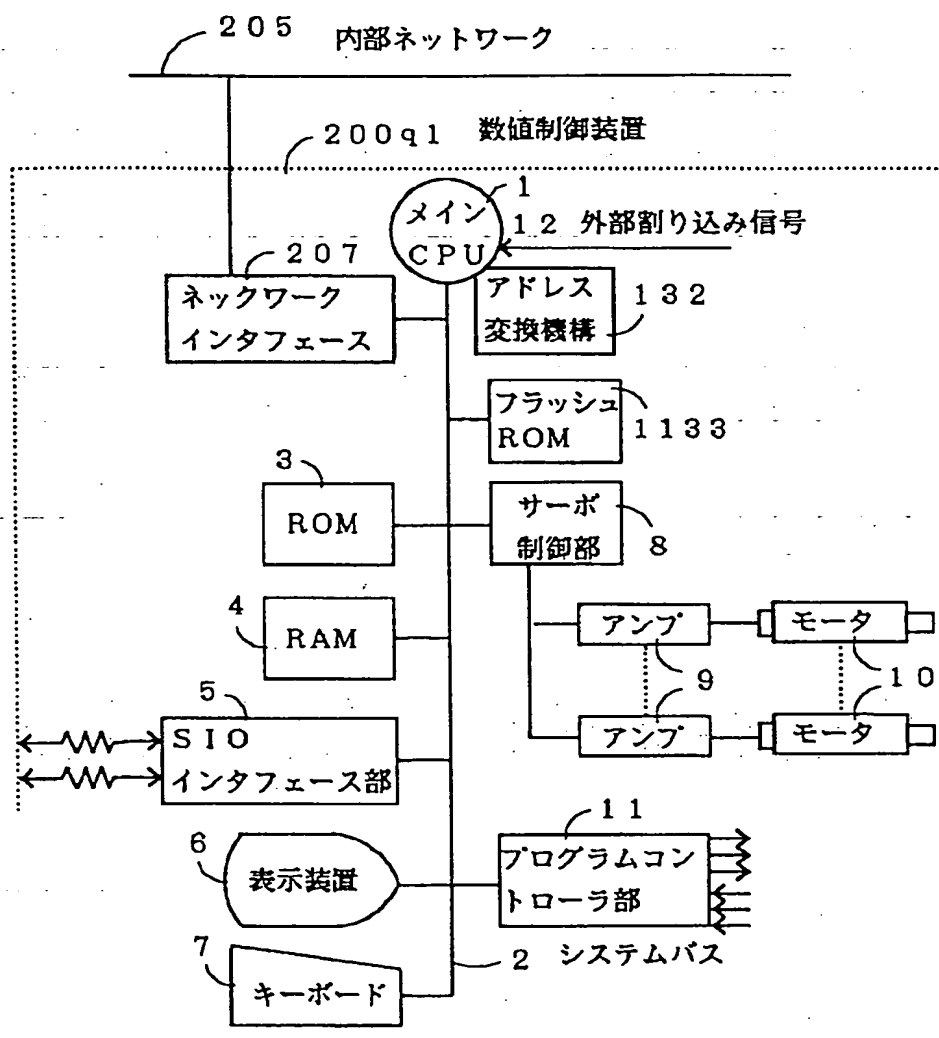
【図24】



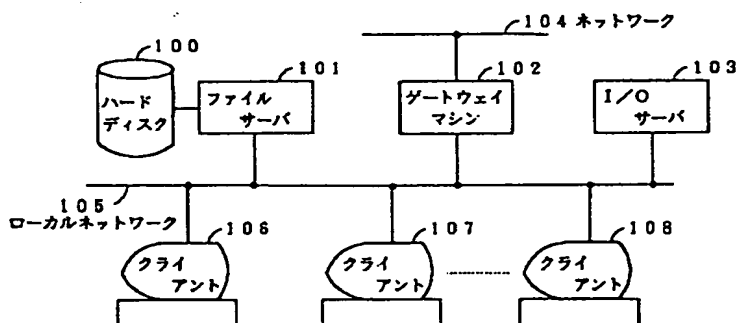
【図33】

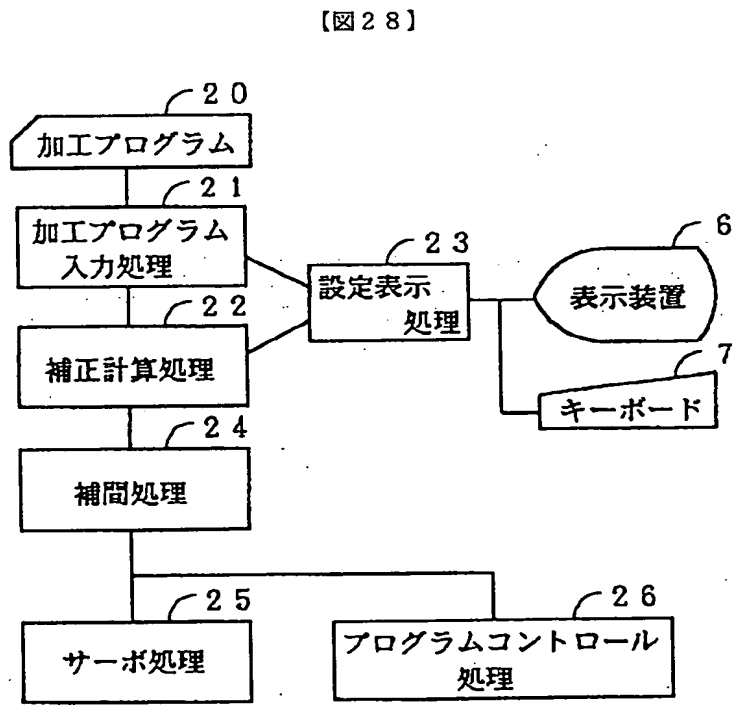
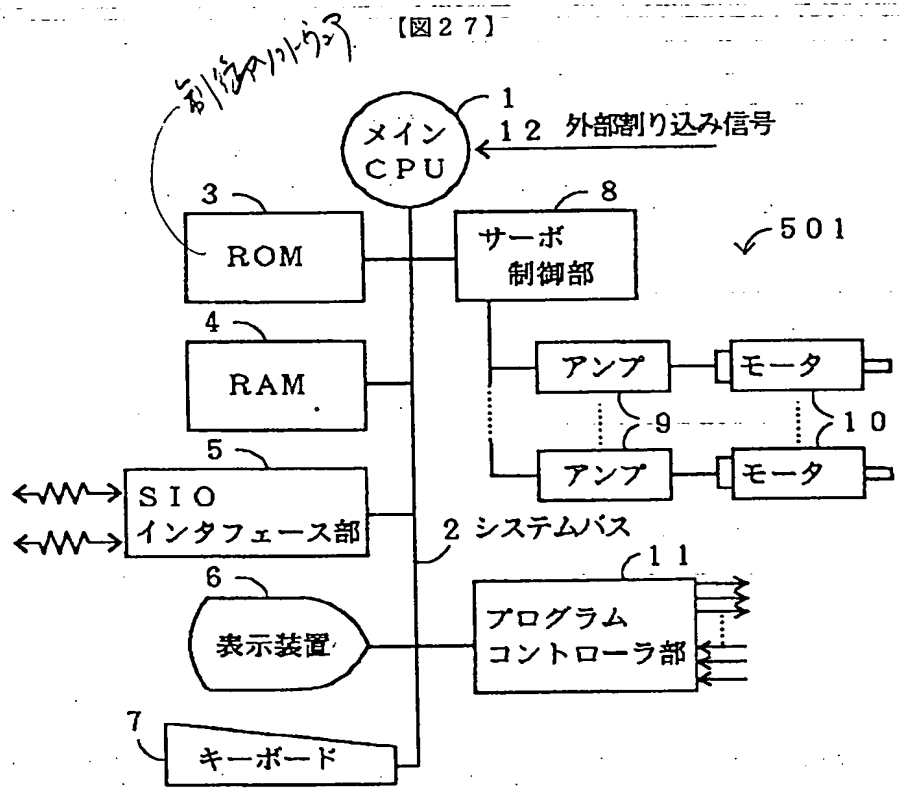


【図26】



【図31】





【図3.2】

